1.0 INTRODUCTION

1.1 Project Overview

In the past, engineers had been designing the engineering systems that requires a lot of hardwares. It is merely impossible to design distance control of the system as more hardwares and wiring were needed. In addition, if engineers wish to improve the design, all the unrelevant hardwares need to be scrap away, which is not environmental friendly, costly and not sustainable.

With the rise of the technology, programmable logic controller (PLC) had ease the engineering design and less materials are needed, as all the design are implemented in software, through programming. PLC had been commonly used in the industry, including controlling induction motor inverter fed variable drive system. Design a distance control machinery is now possible, even by using Ethernet as the communication device between the computer and the PLC.

The convenient part of PLC is, apart of design the program structure by its own propietary software, it is also assessible and controllable by other software provided the software has the driver of the PLC. Therefore engineers can use LabVIEW, which has various types of industries applications that is in virtual instrument (VI) instead of the real and heavy instrument, to control the PLC.

1.2 Project Objectives

The main objective of this project is to implement the Programmable Logic Controller (PLC) by using LabVIEW 2009 via LAN Ethernet for motor control system. The project must achieve the ability on controlling the following aspect:

- Start
- Stop
- Speed Control
- Forward/Reverse Position Control

The start and stop actions can be control directly through the status change of the PLC output. Meanwhile the speed and forward/reverse position control can be controlled by the PLC through the Variable Frequency Drive (VFD) / 3G3MV inverter. All the controlling process must be done by using the main computer that is connected to the PLC, using LabVIEW 2009. The Local Area Network (LAN) is needed in order to have connection between main computer and PLC. Thus, the Ethernet Module is involved in the PLC.

1.3 Thesis Outline

The outline of the thesis is divided into seven main sections as listed below:

Section 1. This is an "Introduction". It introduces the overview and objectives of the thesis. It includes the brief explanation of the project.

Section 2. Is the "Background". This section explaines the overview of PLC and its operation and structure, variable frequency drives (VFD), squirrel cage induction motor, Ethernet, OPC and Human Machine Interface (HMI).

Section 3. It is for the "Project Design Configuration and Preparation", that explaines NI OPC Servers, LabVIEW, the configuration of the PLC Ethernet Unit and the function parameters setting of the VFD.

Section 4. This is the "Project Implementation". It shows the implementation process and LabVIEW virtual instrument design and the description of the program.

Section 5. Is for "Testing and Verification". It shows the testing outcome of the project implementation

Section 6. This section is "Conclusion", summarises the overall project.

Section 7. Is the "Recommendations". It describes the future recommendation that can be implemented in the system demonstrated in this thesis.

2.0 BACKGROUND

2.1 Programmable Logic Controller (PLC)

2.1.1 Overview of PLC

PLC is industrial computer in which the hardware and software had been specially designed to adapt in the industrial environment. PLC can be used to monitor inputs, and depending upon their state make decisions based on its program or logic, to control (turn on/off) its outputs to automate a machine or a process. [1] They are used in many applications such as: material handling, machining, packaging, automated assembly, and countless other industries.

Before the automotive industry discovered the advantages of PLC, the process of modifying relay circuitry was a headache process. In the past, annual car model changes forced plant engineers to constantly modify production equipment managed by relay circuitry. In some cases, the engineers had to scrap entire relay controlled panels and replace them with completely redesigned systems, which are costly, troublesome and not environmental friendly [2]. Now, with the PLC been introduced to the world in 1968 [3], it allows engineers to implement numerous manufacturing changes with relative ease, which reduces changeover costs and downtime. [2]

The first PLCs were introduced in order to eliminating the large cost involved in replacing the complicated relay based machine control systems into US car manufacturer. Bedford Associates (Bedford, MA) proposed something called a Modular Digital

Controller (MODICON) to a major US car manufacturer. The MODICON 084 brought the world's first PLC into commercial production in the mid70's. [4]

The programmable logic controller is becoming a very crucial tool in most industrial automations today as it is rugged and designed to withstand vibrations, temperature, humidity, and noise. Plus PLCs have interfacing for inputs and outputs already inside the controller. In addition, PLCs are easily programmed and have an easily understood programming language [1]. Apart of that, PLC systems also have many other advantages which include [5]:-

- Better and more reliable performance with easy maintenance.
- Much smaller physical size thus easier installation even at very limited area.
- Cost effective for controlling complex systems.
- Flexible and can be changed to other control systems quickly and easily.
- Allowed more complicated controls.
- Reduced downtime with more efficient troubleshooting function and help.

2.1.2 PLC Operation

The PLC operates internally in a way which is identical to normal computers. The inputs are continuously monitored and copied from the I/O module into RAM memory which is divided into the input and output sections. The Central Processing Unit (CPU) steps through the control program in another section of the memory and fetches the input variables from the input RAM. Depending on the program and the state of inputs, the

output RAM is filled with the control variables which are then copied into the I/O module where they control the processes [6]. Figure 2-1 shows the basic PLC operation process.

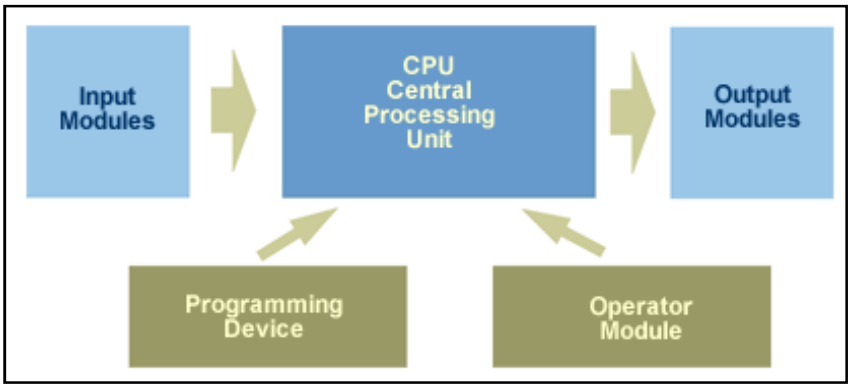


Figure 2-1: Basic PLC Operation Process [7]

2.1.3 PLC Structure

The CPU used in PLC system is a standard CPU present in many other microprocessor controlled systems. The choice of the CPU depends on the process to be controlled. In general 8 or 16 bit CPUs fulfil the requirements sufficiently [6].

The memory in a PLC system is separated into the program memory which is usually stored in EPROM/ROM, and the operating memory. The RAM is essential for the operation of the program and the temporary storage of input and output data. Typical memory sizes of PLC systems are about 1 kb for small PLCs, few kb for medium sizes and more than 10-20 kb for larger PLC depending on the conditions. Many PLC would support easy memory upgrades [6].

Input/Output modules are the boundary between the internal PLC systems and the external processes to be monitored and controlled. Since the PLC is a logic based device

with a typical operating voltage of 5 volts and the external processes usually require higher powers and currents, the I/O modules are optically or otherwise detached. The typical I/O operating voltages are 5V - 240 V dc (or ac) and currents from 0.1A up to several amperes. The I/O modules are designed in this way to minimize or eliminate the need for any intermediate circuitry between the PLC and the process to be controlled. Small PLC units would have around 40 I/O connections with larger ones having more than 128 with both local or remote connections and extensive upgrade capabilities [6].

Programming devices are essential components of the PLC systems. Since they are used only in the development and testing stage of a PLC program, they are not permanently connected to the PLC. The program in a ladder diagram or other form can be designed and usually tested before downloading to the PLC. The Programming devices can be a dedicated device, such as Hand-held Programming Consoles, or a personal computer. It allows the graphical display of the program (ladder diagram). Once the device connected to the PLC, it can read the program and allows for the real time monitoring of its operation to assist debugging. Once the program is found to operate as required the programming device is disconnected from the PLC which continues the operation [6].

2.2 Variable Frequency Drive

2.2.1 Overview of Variable Frequency Drive (VFD)

There are 2 ways to alter the speed of an induction motor, either changing the slip of motor or by changing the frequency of the motor. Slip can be alter depends on the load on the shaft, which also can be controlled by lower down the voltage applied to the motor, or even increasing the rotor resistance in the rotor windings [5]. In this project variable frequency drive will be acting as the speed controller and as well as the inverter.

VFDs are used in many different types of applications for various reasons. For example, they are effective in term of energy saving in pump and fan applications. In addition they enhance process operations, particularly where flow control is involved. VFDs provide soft-start capabilities, which decrease electrical stresses and line voltage sags associated with full voltage motor start-ups, especially when comes to drive high-inertia loads [8]. Figure 2.2 shows the rectifier inverter of VFD.

The drive consists of rectifiers in three phase diode-bridge or in controlled rectifier circuits and inverter switching circuits (most commonly used device is insulated-gate bipolar transistor – IGBT). The rectifiers convert AC input power to DC intermediate power then the inverters will convert the resultant to quasi-sinusoidal AC power as shown in Figure 2-2 [5].

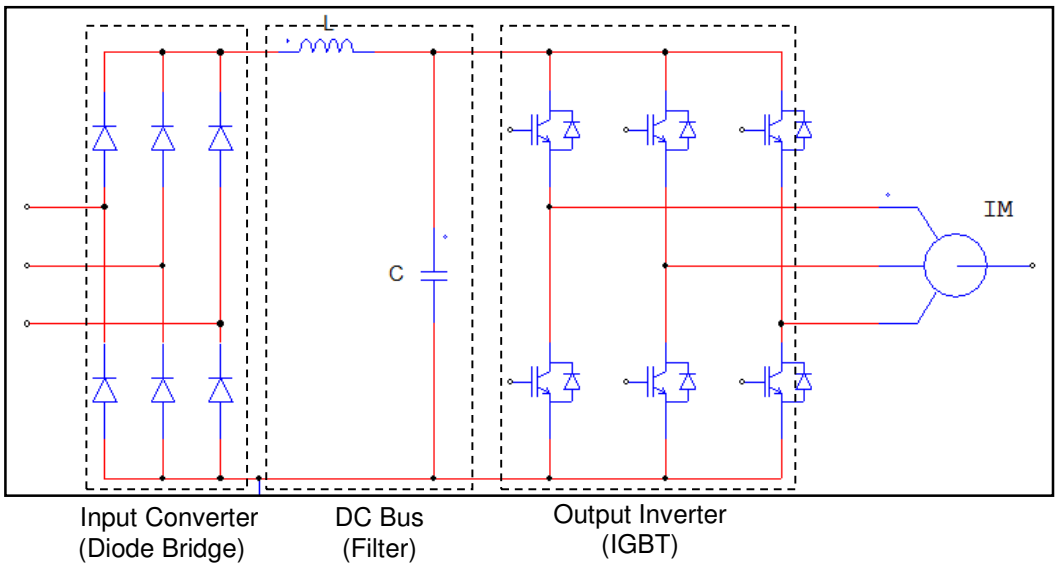


Figure 2-2: Internal Circuit of a Variable Frequency Drive (Sketch using PSIM)

2.3 Squirrel Cage Three Phase Induction Motor

Induction motor is the most common type of three phase alternating current (AC) motor used in the industries today. The reasons are shown below [9]:-

- Cheaper price compared to other types of motors.
- Simple and rough construction.
- More dependable
- High efficiency
- Able to alter the speed easily
- Does not require starter to have starting torque

A three phase induction motor consists of two members, stator and rotor. A stator is the stationary part with winding and a rotor is the moving part with winding or bars of copper. If the rotor is wounded with copper wires it is called wound rotor motors and if the rotor

is constructed of solid copper bars welded together at both ends, the motor is known as squirrel cage rotor motor, figure 2.3 [10].

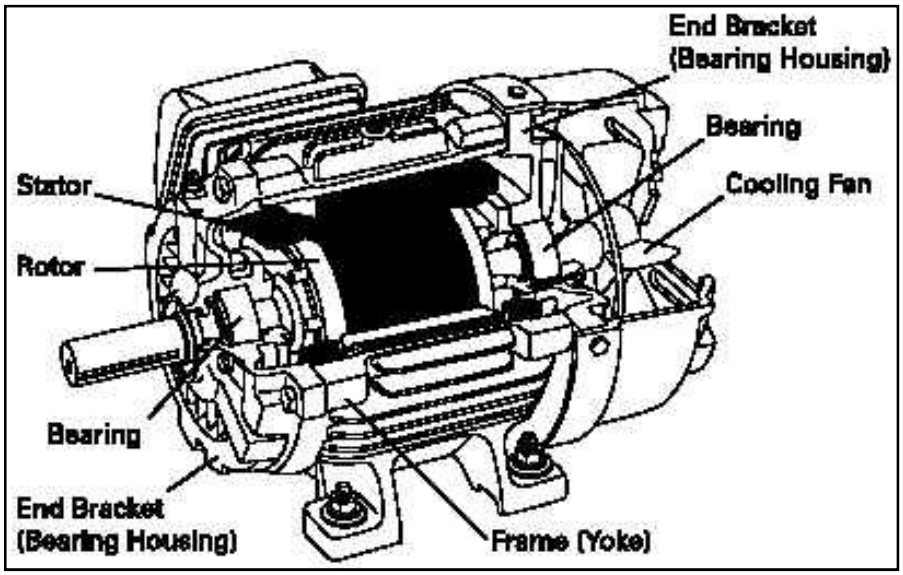


Figure 2-3: Squirrel Cage Three Phase Induction Motor [9]

For squirrel cage induction motor, the stator is a core made up of hundred thin laminations stacked together to form a cylinder and with coils of insulated wire inserted into slots of the core, figure 2.4. As for the rotor, it is constructed of steel laminations piles with evenly spaced aluminium conductor bars cast in the slots of the rotor core around the boundary and welded together at both ends, figure 2.5 [5].

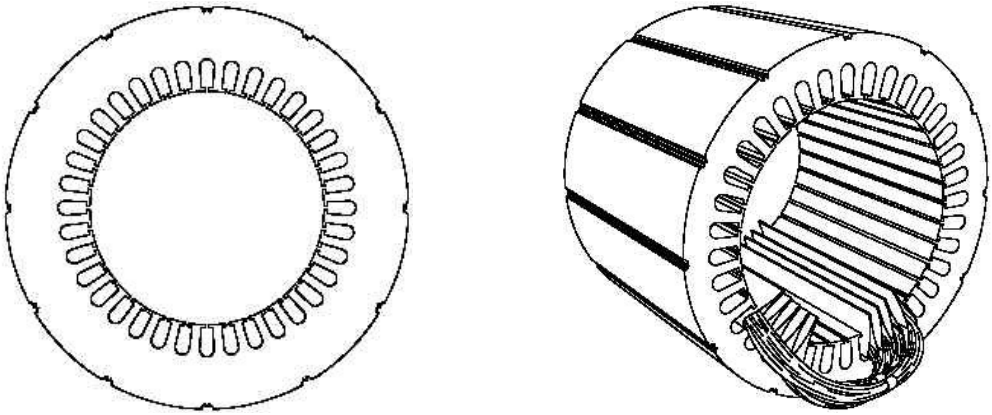


Figure 2-4: Stator Lamination (left) and Stator Construction (right) [9]

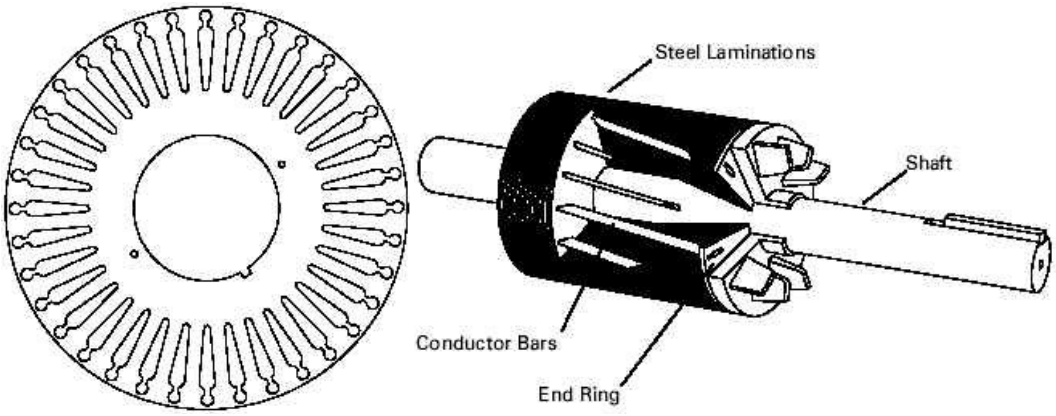


Figure 2-5: Rotor Lamination (left) and Rotor Construction (right)

Stator windings which are positioned around the rotor will generate rotating magnetic field as they move quickly over the rotor when supplied with three phase power. From this process, current is induced in the rotor conductors and when it interacts with the rotating magnetic field by the stator caused the rotating movement of the rotor [5].

The speed of the motor, which is known as the synchronous speed, can be expressed by the following formula [10]:

$$N_s = \frac{120f}{P}\ rpm \quad \text{..... (Equation 1)}$$

Where

N_S = synchronous speed of the motor

f = frequency of the power supply

P = number of poles of the motor

Equation 1 shows that the speed of the motor is proportional to the frequency of the power supply, which explains the concept at the output part of the project i.e. controlling induction motor by altering the frequency of the power supply.

However, according to the load torque, induction motor never runs at synchronous speed but slightly less. The difference in these two speeds is the 'slip' of the motor usually expressed in percent and is given by:

$$S = \frac{N_S - N_r}{N_S} \qquad \text{..... (Equation 2)}$$

Where

S = slip

N_S = synchronous speed of the motor

N_r = rotating speed of the motor

2.4 Ethernet

2.4.1 Overview of Instrumentation Networking

Communication is one of the essential tools in monitoring and control system as the data transfers between the control units and the devices such as sensors creates the objectives of the system. The link between the devices and softwares are named as network, and the process of the communication named as networking.

Networking of hardware and software resources is crucial for [11]:

- Bringing multiple devices together for exchange of information and joint operations.
- Sharing of functions of equipment and devices.
- Data sharing, which allows a group of users to exchange information among each other on periodical or regular basis
- Routing data from one network element to another
- Coordination of activities of equipment and devices in plant and production
- Monitoring the status of equipment and devices
- Monitoring and taking corrective actions on safety issues
- Production planning, configuring, and reconfiguring

There are three types of networks, which are [11]:-

- **Wired networks**: devices are connected by wires hence fixed in space.
- **Wireless networks**: devices are connected wirelessly hence can move in space, and

- **Hybrid networks**: devices are connected in which both wired and wireless connections are used in combination.

Meanwhile data is collection of alphabetic, numeric, or special function characters that are appropriately grouped in binary format to form words, messages, or information. Data is measured in baud rates or bit rates. Baud rate specify the number of symbols transmitted in a unit time or per seconds. While bit rate specify the number of bits transmitted per second. Baud rates and bit rates are the same if only one bit is assigned per symbol [11].

Digital information is generally transmitted in data frames. Data frame is collection of characters, Figure 2-6, that convey a complete message that can be understood by the transmitting and receiving devices [11].

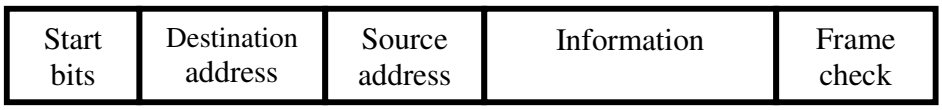


Figure 2-6: Digital data transmission in frames [11]

When data frames are used, the information rate is not the same as bit rates or bauds since it contains overhead data addresses, error checks, start and stop information, and so on. The type of information in the frames is governed by protocols and standards. [11]

Data from one device to other are transmitted in serial or parallel forms [11] Figure 2-7 and 2-8 explaine the difference between the serial and parallel data transmission.

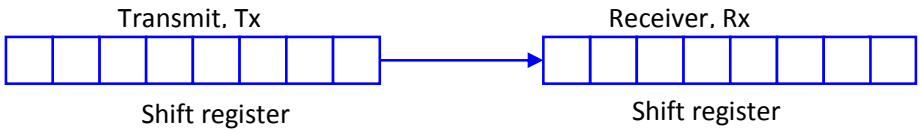


Figure 2-7: Serial Data Transmission [11]

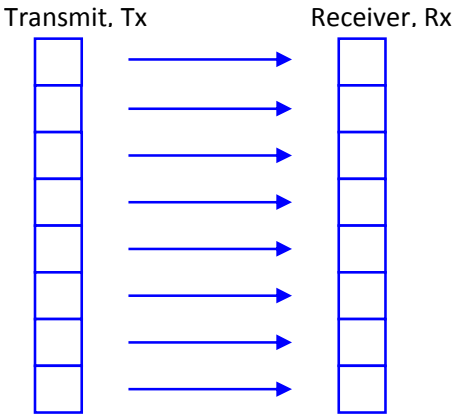


Figure 2-8: Parallel Data Transmission [11]

In general there are 3 modes of channel operation [11]:-

- **Simplex operation**: transmission can take place only in one direction from one device to another.
- **Half-duplex operation**: transmission can take place in either direction, but only one direction at a time.
- **Full-duplex operation**: transmission takes place in both directions simultaneously.

Figure 2.9 expresses the 3 modes

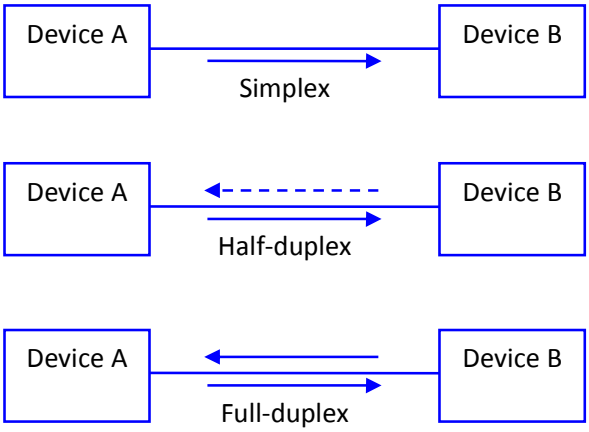


Figure 2-9: Three Modes of Channel Operations

In networks, the transmission is recognized by using various channels. A channel may be defined as a single path on a line trough which signals flow. Lines are defined as the components and parts that extend between the terminals of the communicating devices [11].

There are a variety of forms of networks depending on the number of network elements and their spatial distributions. Local area network (LAN) is a system for interconnecting data communicating components within a comparatively detain space. While wide area networks (WANs) extent into greater geographic distances linking two or more separate LANS [11]

There are many other terms used apart of LANs and WANs such as metropolitan area networks (MANs), personal area networks (PANs), and so on, but these are basically LANs or WANs with varying sizes. The major point of networks is to share resources by connecting network elements called nodes. In order to connect nodes, four elements are

needed; 1) the transmission medium, 2) the network topology, 3) the protocols, and 4) the network operating system [11].

Transmission medium can be defined as the physical path between nodes of the network. Physical path may be cables, fibres, radiofrequency devices, microwave devices, etc. Meanwhile topology refers to physical layout of devices. The network topology can have a significant effect on the performance and efficiency of the network and growth potential. Whereas protocols are the set of rules that are agreed to enable communication between devices. Whilst operating systems are the software running on the background managing the sharing of equipment and data between the network nodes [11]

Communication process between two devices A to B is illustrated in Figure 2-10 [11].

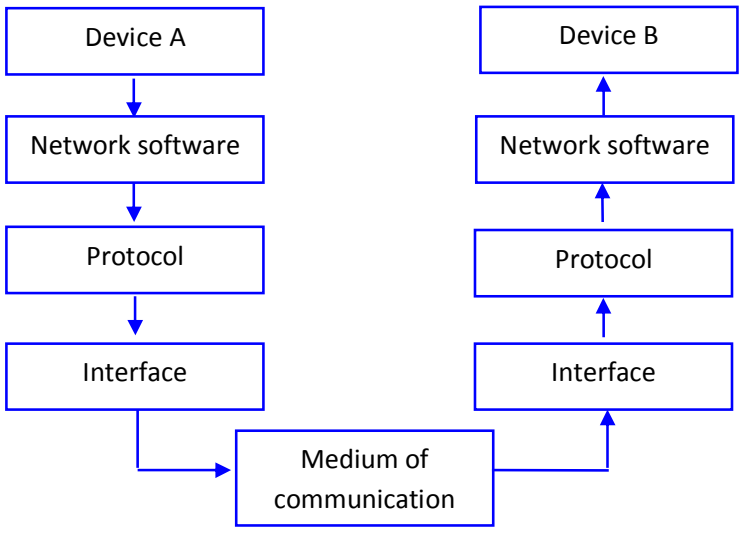


Figure 2-10: Process of communication of networked devices [11]

2.4.2 Overview of Protocols

Protocol is a set of rules that are agreed by related authorities in order to allow successful communications between devices. Protocols are created according to the Open System Interconnection (OSI) reference model. The seven layers of OSI reference model are illustrated in Figure 2-12 [12].

OSI Layer	Name		Common Protocols	
7	Application			
6	Presentation		HTTP | FTP | SMTP | DNS	
5	Session			
4	Transport		TCP	SPX
3	Network		IP	IPX
2	Data Link		Ethernet	
1	Physical			

Figure 2-11: Seven Layers of OSI Reference Model [12]

The seven layers of the OSI reference model can be divided into lower layers (1–4) and upper layers (5–7). The lower layers focus on data-transport functions while the upper layers focus on the applications [13].

- **Level 1 - Physical Layer:**

  This layer divides the transmission data into frames and manages to make it ready for the interface to the network medium. A typical example of a physical layer protocol and hardware is the RS-23 [12].

- **Level 2 - Data Link Layer:**

  Outgoing messages are pulling together into data frames, and acknowledgements from the receivers are scheduled for each message transmitted. Data integrity is

checked and error-detecting and error-correcting codes are issued. There are many examples of hardware and software operating at this level, such as the Binary Synchronous Communications Protocol (BISYNC), X.25 [12].

- **Level 3 - Network Layer:**

  This layer deals with addressing of the messages for delivery and converts logical network addresses into their physical counterparts. Physical network addresses are known as the Media Access Control (MACs) addresses. MAC decides how to rout the transmission from sender to receiver. It deals with packet switching, data routing, and congestion control of the network [12].

- **Level 4 - Transport Layer:**

  This layer manages the transportation of data from sender to receiver across the network. This layer is crucial to ensure the flow control by making sure that the recipient of the transmitted data is not overlapped with the capacity of the data that it can handle. Long data payloads are fragmented into fractions matching the maximum packet size that can be suitable by the network medium and the recipient [12].

- **Level 5 - Session Layer:**

  This layer set up the system-to-system connection across the network and letting two parties to hold ongoing communications, called a session. This layer has many roles such as: setting up the session, monitoring session identification

process, offering security in data flow, continuity in exchanging data and messages, and terminating the session [12].

- **Level 6 - Presentation Layer:**

  This layer deals with data formatting. For the outgoing messages, it converts data into generic formats so that they can pass through the transmission [12].

- **Level 7 - Application Layer:**

  Application layer is the top layer of the OSI reference model. This is the level that can be seen by the individual users and it provides a set of interfaces for applications in order to gain access to network services. At this layer, the network transparency is maintained by hiding the physical distribution of resources from the users.

However, the conventional OSI model is more likely an academic guideline as the real world applications have the combination of multiple OSI layers into a single layer. The commonly used alternative model in the networking industry is the TCP/IP model. Figure 2-12 illustrates the TCP/IP architecture [14]:

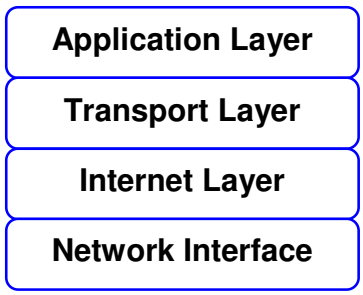


Figure 2-12: TCP/IP Model

- **Level 1 – Application Layer:**

  This layer combines the application, session and presentation layer of the OSI model. Examples of these layers are e-mail, remote login, network management and file transfer protocol (FTP) [14]

- **Level 2 – Transport Layer:**

  This layer is identical to the transport layer in the OSI model. It has two types of protocol which are Transmission Control Protocol (TCP) and User Datagram Protocol (UDP). TCP is connection oriented, i.e. the receiving device allows to send byte stream data or to send data in an orderly form. This allows the retransmission of the corrupted data on upon the request from the receiver. On the other hand, UDP is a connectionless transfer of individual messages and provides no mechanism or error recovery and flow control. This makes UDP less reliable and more suitable for quick data transfer that does not contain secure files [14].

- **Level 3 – Internet Layer**

  This layer corresponds to the network layer of the OSI model and uses the Internet Protocol (IP) to transfer data packets. In this layer, transfers of information across networks are performed via gateways and routers. Network failure can be rectified around the point of failure without correcting the overall system connection

- **Level 4 – Network Interface Layer**

  This layer combines the physical and data link layer of the OSI model and manages the exchange of data between networks via routers and gateways

2.4.3 Overview of Ethernet

Ethernet is the hardware part of a network which acts as a medium for frames to travel. Ethernet is a widely installed local area network (LAN) technology [14]. It was formerly developed by Xerox from an earlier specification called Alohanet (for the Palo Alto Research Center Aloha network) and then developed further by Xerox, DEC, and Intel. An Ethernet LAN typically uses coaxial cable or special grades of twisted pair wires [15]

Ethernet was named by Robert Metcalfe from Xerox, for the passive substance called "luminiferous (light-transmitting) ether" that was previously thought to spread through the universe, carrying light throughout. Ethernet was named in this way in order to describe the way that cabling, also a passive medium could equally carry data everywhere throughout the network [15]

According to Cisco Systems, Ethernet is the favorite protocol to be used because:

- Relatively easy to understand, implement, manage, and maintain
- Allows low-cost network implementations
- Provides extensive topological flexibility for network installation
- Guarantees successful interconnection and operation of standards-compliant products, regardless of manufacturer

There are two types of operation in the Ethernet standard, which are half-duplex and full-duplex modes. In the half duplex mode, data are transmitted using the popular Carrier-Sense Multiple Access/Collision Detection (CSMA/CD) protocol on a shared medium. The main weaknesses of the half-duplex are the efficiency and distance limitation, in which the link distance is limited by the minimum MAC frame size. This limitation reduces the efficiency severely for high-rate transmission. Therefore, the carrier extension technique is used to ensure the minimum frame size of 512 bytes in Gigabit Ethernet to attain a reasonable link distance [16]

In today's technology, there are 4 types of Ethernet [13,16]:-

- **10 BASE-T Ethernet (IEEE 802.3)**

  This is a traditional type of Ethernet which delivers performance of up to 10 Mbps over twisted-pair copper cable.

- **Fast Ethernet (IEEE 802.3u)**

  Fast Ethernet delivers a speed which is 10 times the 10BASE-T Ethernet specification (100 Mbps) while preserving many of Ethernet's technical specifications. These similarities allow organizations to use 10BASE-T applications and network management tools on Fast Ethernet networks.

- **Gigabit Ethernet (IEEE 802.3z)**

  This Ethernet delivers a speed which is tenfold over Fast Ethernet to 1000 Mbps, or 1 Gbps. Because it is based upon the current Ethernet standard and compatible

with the installed base of Ethernet and Fast Ethernet switches and routers, network managers can support Gigabit Ethernet without need to retrain or learn a new technology.

- **10 Gigabit Ethernet (IEEE 802.3ae)**

  10 Gigabit Ethernet endorsed as a standard in June 2002, in which it is an even faster version of Ethernet. It can support intelligent Ethernet-based network services, interoperate with existing architectures, and minimize users' learning curves. Its high data rate of 10 Gbps makes it a good solution to deliver high bandwidth in WANs and metropolitan area networks (MANs).

In general, the Ethernet system consists of three basic elements [16]:

- The physical medium used to carry Ethernet signals between computers.
- A set of medium access control rules embedded in each Ethernet interface that allow multiple computers to fairly arbitrate access to the shared Ethernet channel.
- An Ethernet frame that consists of a standardized set of bits used to carry data over the system.

As with all IEEE 802 protocols, the ISO data link layer is divided into two IEEE 802 sub layers, the Media Access Control (MAC) sub layer and the MAC-client sub layer. The IEEE 802.3 physical layer corresponds to the ISO physical layer [16]. Figure 2.13 shows the relation between Ethernat and ISO/IEEE802

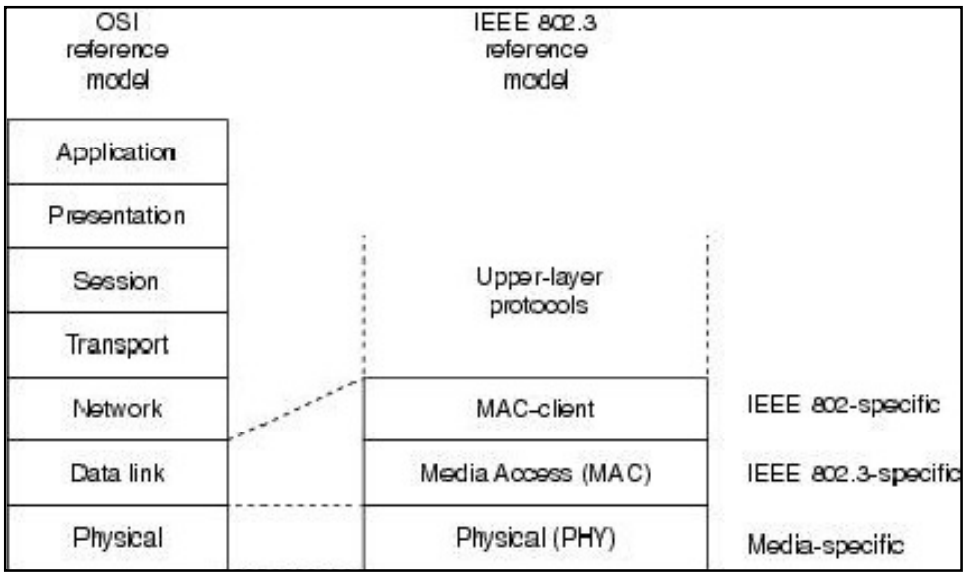


Figure 2-13: Ethernet logical relationship to the ISO reference model

The configuration of cables, computer and other peripherals are defined as the physical topology of a network. In general there are 3 types of physical topologies [17]:

- Linear Bus
- Star
- Tree or Expanded Star

The star-connected topology is the most commonly used Ethernet configuration. The Figure 2-14 illustrates the network configuration.

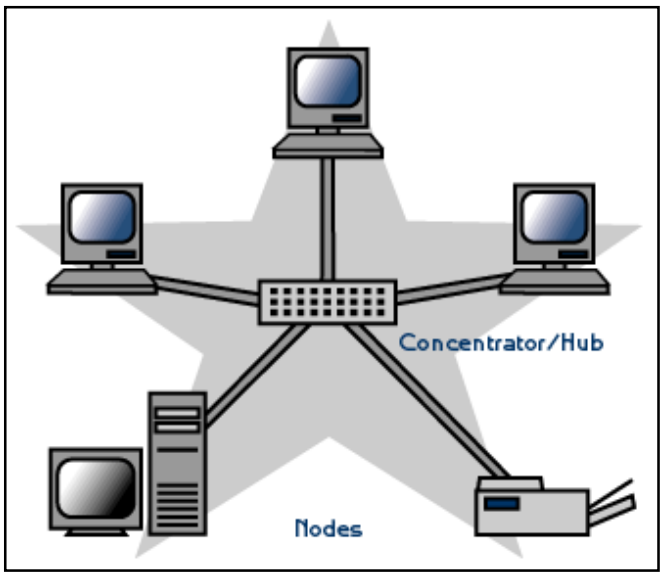


Figure 2-14: Star-connected topology [17]

A star-connected topology is designed in the way that with each node (example file server, workstations, and peripherals) connected directly to a central network hub, switch, or concentrator [17]

Data on a star network passes through the hub, switch, or concentrator before continuing to its destination. The hub, switch, or concentrator manages and controls all functions of the network. It also serves as a repeater for the data flow. This configuration is common with twisted pair cable; however, it can also be used with coaxial cable or fibre optic cable [17]

The IEEE 802.3 standard also states the data frame format that is required for all MAC implementations including optional formats that are used to expand the basic capability of the protocol. The 7-field data frame format structure is illustrated in Fig. 2-15.

7	1	6	6	2	46-1500bytes	4
Pre	SFD	DA	SA	Length Type	Data unit + pad	FCS

Figure 2-15: IEEE 802.3 data frame format

Fig 2-16 illustrates the MAC frame for the transmission rates with Gigabit carrier extension field.

7	1	6	6	2	Variable	4	Variable
Pre	SFD	DA	SA	Length Type	Data unit + pad	FCS	Ext

Figure 2-16: MAC frame with Gigabit carrier extension

2.5 IP Address

Every machine interfaced to a router or host in a network has a unique identifying number, called an IP address. IP Address is the acronym for Internet Protocol address. [14,18] Example below illustrates how IP address looks like:

192.168.1.45

An IP address is 32 bits long, in another words it has 2^{32} possible combinations of IP addresses. An IP address is the combination of 4 decimal notations separated by a period. Each decimal can have a number in the range of 0 to 255. IP addresses are written in decimal notation so that it is readable by humans but its actual form is binary as computers can only read information in binary notation. For example, the given IP address above has a binary notation given in Figure 2-17 [14].

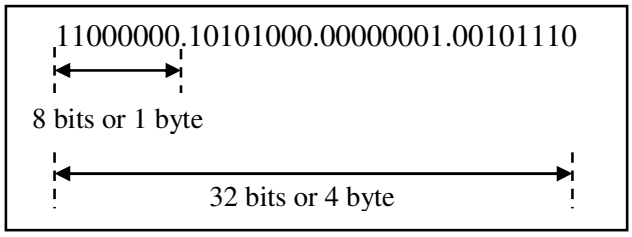


Figure 2-17: Binary representation of an IP address

Each section has an 8 bit decimal representation (equivalently 1 byte or an octet) where the minimum can be 0 (2^0) and the maximum number can be 255 (2^8). The above example is a representation of an IPv4 address. There are other types of IP address versions available such as IPv6 anticipated to compensate for the exhaustion of IPv4 address, i.e. running out of space for IPv4 address. However, it is not be included in this thesis as the project uses an IPv4 type address [14]

There are 5 IP classes plus certain special address, illustrated in Table 2-1:

Table 2-2: Five types of IP Classes and Special Addresses [18]

Class	Description
Default Network	The IP address of 0.0.0.0 is used for the default network.
Class A	This class is for very large networks, such as a major international company might have. IP addresses with a first octet from 1 to 126 are part of this class. The other three octets are used to identify each host. This means that there are 126 Class A networks each with 16,777,214 (2^{24} – 2) possible hosts for a total of 2,147,483,648 (2^{31}) unique IP addresses. Class A networks account for half of the total available IP

	addresses. In Class A networks, the high order bit value (the very first binary number) in the first octet is always 0. **Net** **Host or Node** 115 24.53.107
Loop Back	The IP address 127.0.0.1 is used as the loop back address. This means that it is used by the host computer to send a message back to itself. It is commonly used for troubleshooting and network testing.
Class B	Used for medium-sized networks. A good example is a large college campus. IP addresses with a first octet from 128 to 191 are part of this class. Class B addresses also include the second octet as part of the Net identifier. The other two octets are used to identify each host. This means that there are 16,384 (2^{14}) Class B networks each with 65,534 (2^{16} – 2) possible hosts for a total of 1,073,741,824 (2^{30}) unique IP addresses. Class B networks make up a quarter of the total available IP addresses. Class B networks have a first bit value of 1 and a second bit value of 0 in the first octet. **Net** **Host or Node** 145.24. 53.107
Class C	This class is commonly used for small to mid-size businesses. IP addresses with a first octet from 192 to 223 are part of this class. Class C addresses also include the second and third octets as part of the Net identifier. The last octet is used to identify each host. This means that there are 2,097,152 (2^{21}) Class C networks each with 254 (2^8 – 2) possible hosts for a total of 536,870,912 (2^{29}) unique IP

	addresses. Class C networks make up an eighth of the total available IP addresses. Class C networks have a first bit value of 1, second bit value of 1 and a third bit value of 0 in the first octet. **Net** **Host or Node** 145.24. 53. 107
Class D	Used for multicasts, Class D is slightly different from the first three classes. It has a first bit value of 1, second bit value of 1, third bit value of 1 and fourth bit value of 0. The other 28 bits are used to identify the group of computers the multicast message is intended for. Class D accounts for $1/16^{th}$ (268,435,456 or 2^{28}) of the available IP addresses. **Net** **Host or Node** 224 24.53.107
Class E	Used for experimental purposes only. Like Class D, it is different from the first three classes. It has a first bit value of 1, second bit value of 1, third bit value of 1 and fourth bit value of 1. The other 28 bits are used to identify the group of computers the multicast message is intended for. Class E accounts for 1/16th (268,435,456 or 2^{28}) of the available IP addresses. **Net** **Host or Node** 240 24.53.107
Broadcast	Messages that are intended for all computers on a network are sent as broadcasts. These messages always use the IP address 255.255.255. 255

2.6 OLE for Process Control (OPC)

OLE for Process Control (OPC), which is the short form for Object Linking and Embedding for process control, is a standard interface between various data sources, including devices on a factory floor, laboratory equipment, test system fixtures, and databases in a control room [19].

OPC allows device side server and application software, which are two separate processes, to communicate with each other through Microsoft COM/DCOM interface. COM is abbreviation for Component Object Model and meanwhile DCOM is Distribution COM. COM provides an interface communication layer that allows local and remote procedure calls to be made between processes. DCOM is the natural extension of COM to support communication among objects on different computers, i.e. on a LAN, WAN, or the Internet [19].

OPC was designed to be a layer of abstraction between the specific device and the program that needs to acquire information or control that device. The OPC standard specifies the behaviour that the interfaces are expected to provide to the clients who use them, and the client receives the data from the interface using standard function calls and methods. As a result, any computer analysis or data acquisition program can communicate with any industrial device as long as that program has an OPC client protocol and the device driver has an OPC interface associated with it [19].

2.7 Human Machine Interface (HMI)

Human machine interfaces (HMI) are graphical representations of a system that consist of system configurations, alarms, data collection and trend logging capabilities. The HMI is usually hosted in the computers of the master station where it can remotely monitor and control processes and field devices with the click of a button. HMIs can also be configured to remotely monitor and control plant processes or field devices from other computers within a plant Ethernet network or the Internet. However, it is advisable that user monitoring and control permissions as well as control time granted to the user(s) to be restricted to a certain period and group. This is to ensure that control is given to the appropriate user to avoid unauthorized action that can affect the system's operability [14].

3.0 PROJECT DESIGN CONFIGURATION & PREPARATION

3.1 NI OPC Server

NI OPC Server was used in this project as it has OMRON FINS Ethernet driver that allow the communication between OMRON CJ1M-CPU11-ETN21 PLC with LabVIEW. OMRON originally supplies their customers with FINS gateway, interfacing software that communicates with the PLC and its proprietary software, OMRON CX-Programmer over the Ethernet network [14].

With the OMRON FINS Ethernet driver in NI OPC, users can setup the server by just a few simple setups and create variable tags that can be links directly to the PLC's registers. These tags are named as OPC tags. The NI OPC Servers also have NI OPC Quick Client that enable users to monitor the status of the PLC in real-time [14].

As long as the OPC tags had been created, the communication between the LabVIEW and PLC had been simplified as the driver can automatically apply the relevant FINS commands provided the tags are correctly configured [14]. Meanwhile in LabVIEW, the program can be design by using Shared Variables which is link to the OPC tags.

Figure 3-1 shows the layout of both NI OPC Servers and NI OPC Quick Client.

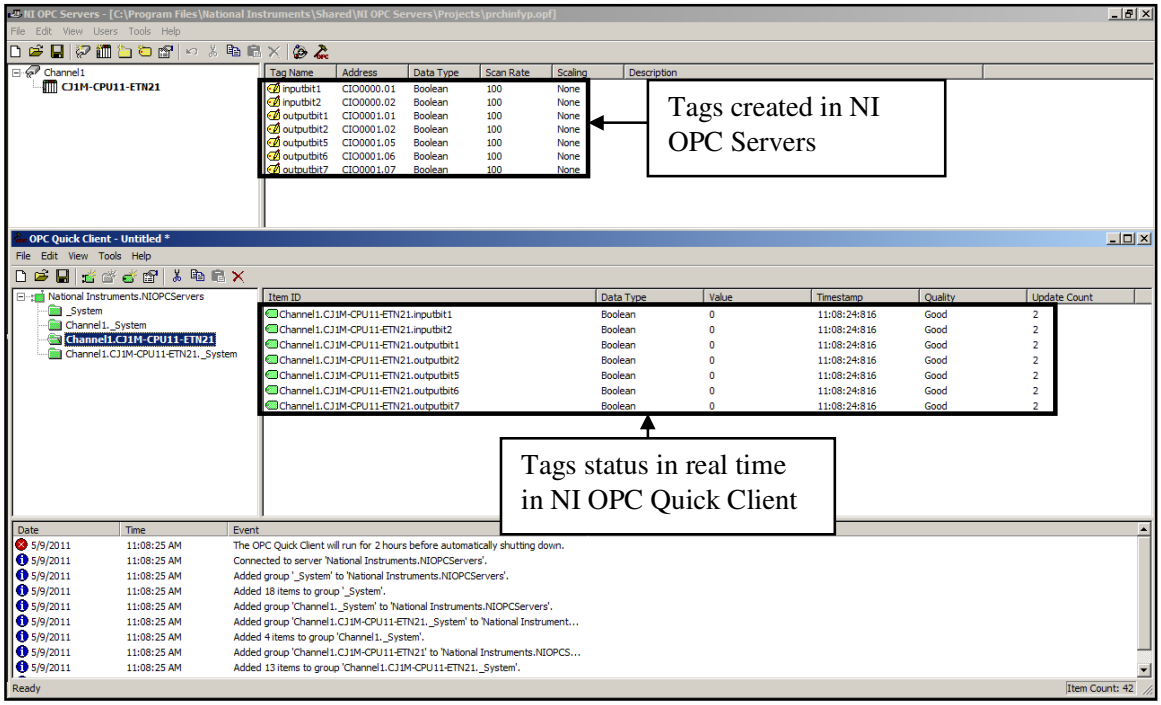


Figure 3-1: NI OPC Servers with NI OPC Quick Client

The setup and configuration of NI OPC Servers is demonstrated in Appendix B.

3.2 LabVIEW

3.2.1 Overview

LabVIEW is the acronym for Laboratory Virtual Instrumentation Engineering Workbench and is a graphical development environment for generating flexible and scalable design, control, and test applications rapidly at minimal cost. With LabVIEW, engineers and scientists are able to interface with real-world signals, analyse data for meaningful information, and share results through intuitive displays, reports, and the Web. Regardless of programming experience, LabVIEW makes development fast and easy for all users [14,20].

3.2.2 Programming Style

The programming style used in LabVIEW is G programming, which is abbreviation for graphical programming. It is also known as dataflow programming as it is depending on the structure of the graphical block diagram to execute the user-designed program. Compared to text-based programming, LabVIEW is user friendly as the user can design the program by simply arrange and wiring the relevant icons together [14,20]. LabVIEW programs are named as virtual instruments, or VI, because their appearance and operation mimic the physical instruments, such as oscilloscope and multimeters [21]. Similar to other conventional programming, LabVIEW has standard features such as looping structures, data structures, event-handling, object-oriented programming. LabVIEW also has an extensive library of math functions similar to MATLAB libraries and also formula

nodes that allow text-based programming for certain sections of the code that require complex logical structures. Besides that, LabVIEW also has networking library functions that can easily allow users to reference [14].

Compared to other softwares like Microsoft Visual Basic, LabVIEW is a better option as it comes together with a library of functions included Shared Variables Project Library, which is bound to the OPC tags, that allow server and client communication by connecting relevant icons with the Shared Variables. If Microsoft Visual Basic (VB) is used, the OMRON FINS Ethernet driver must be developed using the MS Comm function and this would require more time to develop the code [14,22].

LabVIEW has front-end interface applications that allow user to design and then use for controlling application systems. In general LabVIEW has three main elements: the front panel, the block diagram and the connector panel. The front panel allows the user to build the controls and indicators. The controls are including knobs, push buttons, dials, and other input mechanism. Indicators are graphs, LEDs, and other output display. Meanwhile the block diagram let user to add code using Vis and structures to control the front panel objects. The connector panel allows user to represent a single VI as a sub VI icon that can be called in another VI. The elements are illustrated in Figure 3-2.

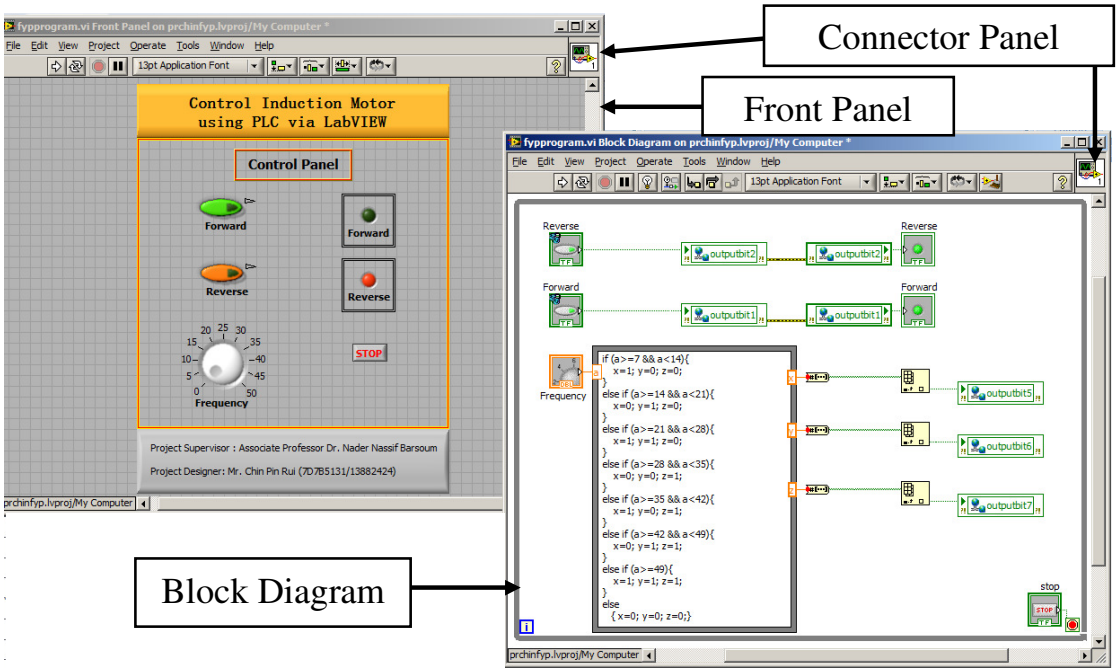


Figure 3-2: Three main elements of LabVIEW software

3.2.3 Shared Variables Project Library

Shared variable is a library function variable that allows sharing of data between applications or different data sources across a network. There are many existing data sharing method in LabVIEW, such as UDP/TCP, LabVIEW queues, and Real-Time FIFO. Compared to Datasocket Communication, using shared variable is user friendly as the configuration can be done by simple setup in the library instead of writing URL and perform more wiring in Datasocket Communication [14,23].

The demonstration of setting up shared variable in LabVIEW is further explained in Appendix C.

3.3 Programmable Logic Controller (PLC)

3.3.1 PLC Setup

The PLC used in this project is OMRON CJ series. There are 4 units used in this PLC, which are

- Power Supply Unit
- CPU Unit with Ethernet function
- Basic Input Unit
- Basic Output Unit

Figure 2-3 shows the dimension of the Power Supply Unit, which is CJ1W-PA202. Meanwhile Figure 2-4 shows the dimension of the CPU Unit, which is CJ1M-CPU11-ETN21. Then Figure 2-5 shows the dimension of the Basic Input and Output Unit, which are CJ1W-ID211 and CJ1W-OC211 with 18 terminal blocks.

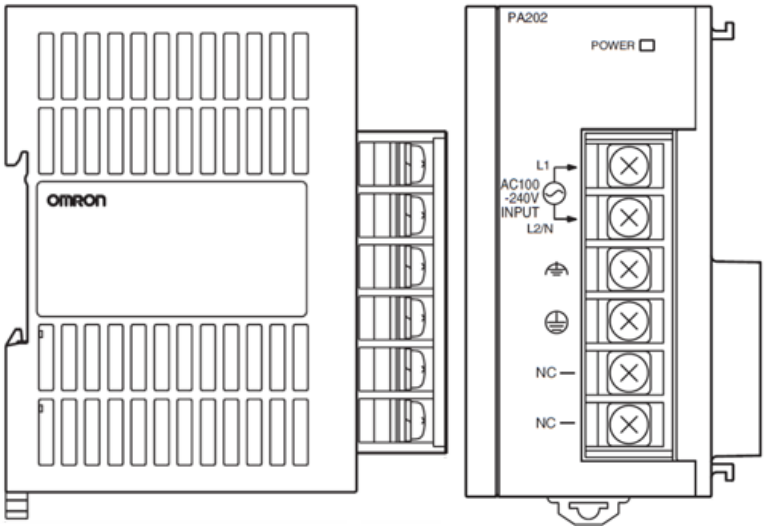


Figure 3-3: Dimension of CJ1W-PA202 Module [24]

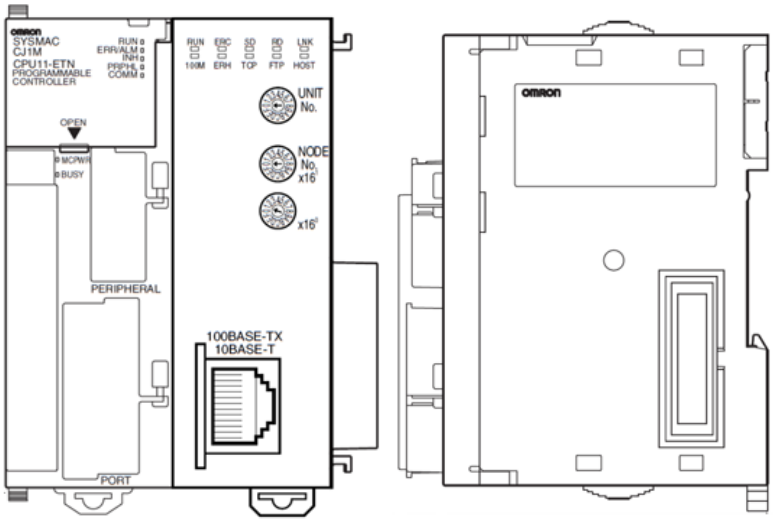


Figure 3-4: Dimension of CJ1M-CPU11-ETN21 Module [25]

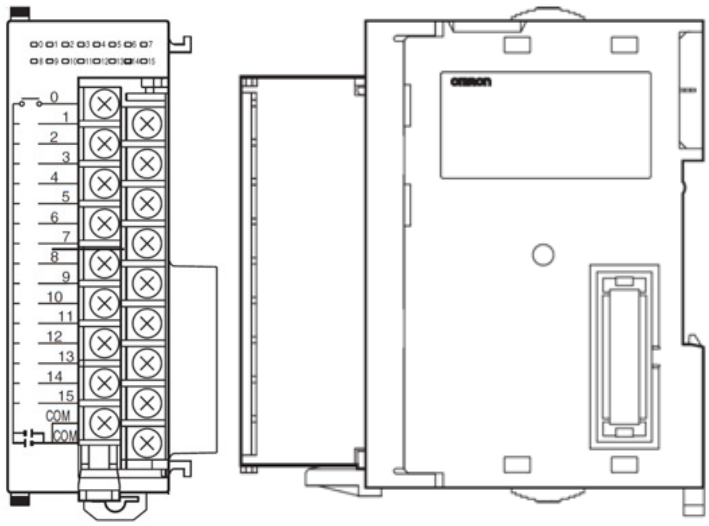


Figure 3-5: Dimension of the ID211 and OC211 (18 terminal blocks) [24]

There is an End Cover for the PLC unit. All these units can be connected by assembled them together and lock the sliders by moving them towards the back of the units. The End Cover must be connected on the far right hand side of the PLC. Otherwise fatal error will occur. Figure 3-6 demonstrate the typical connection of the PLC units [5].

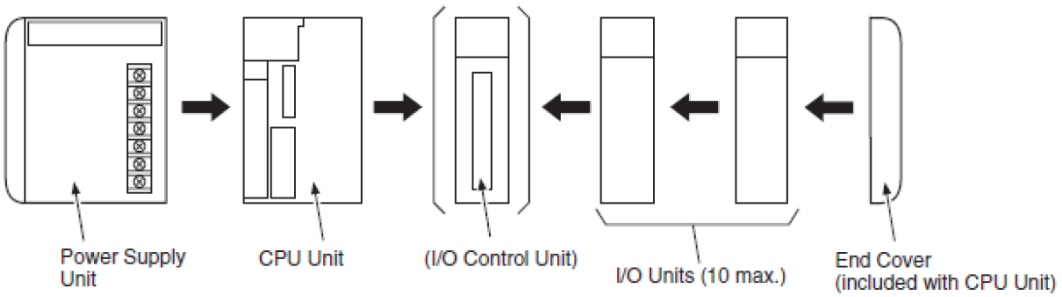


Figure 3-6: Typical Connection of PLC Units [5]

Figure 3-7 demonstrate the arrangement of the PLC units in this thesis.

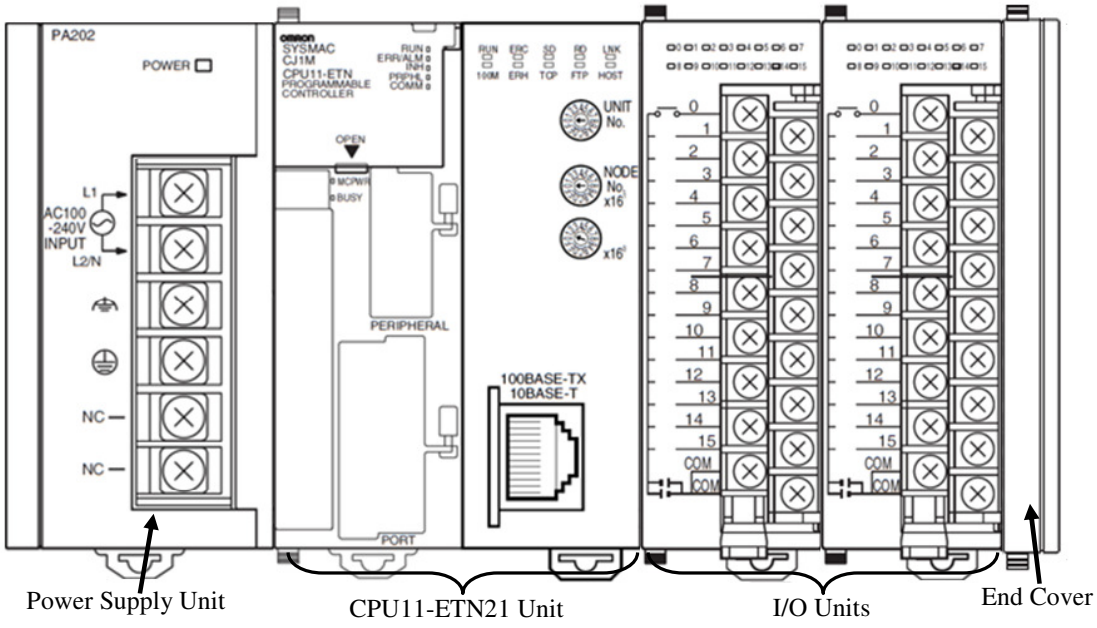


Figure 3-7: Connection of PLC Units for CJ1M-CPU11-ETN21 with I/O Units

3.3.2 CJ1W-ETN21 Hardware Configuration

In order to let the PLC to operate through the Ethernet, the PLC must be given an IP address with a destination node number (DA1) that is not a duplicate of DA1s of other IP addresses in the network. The destination node number is also known as the last octet of an IP address. The PLC node number is configured by turning the dials on the CJ1W-ETN21 module as illustrated in Figure 3-8 the PLC must also be given a unit number in the network besides a destination node address (DNA) [14].

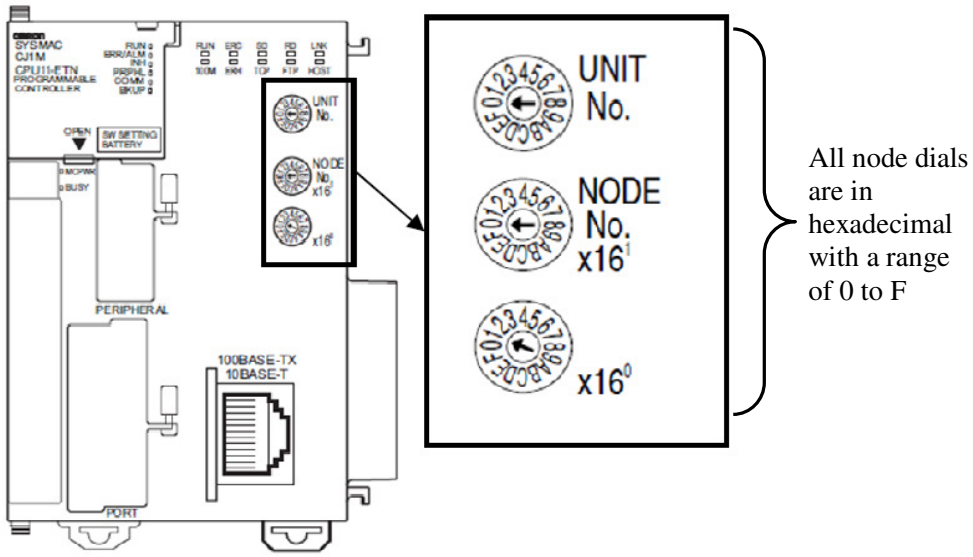
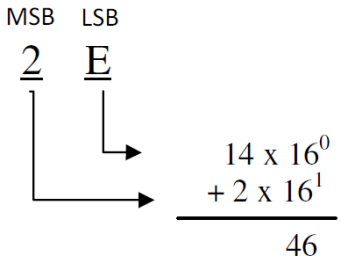


Figure 3-8 CJ1W-CPU11-ETN21 Ethernet Module Hardware Configuration

Node No x 16^1 is the most significant bit (MSB) of the node address whereas $x16^0$ is the least significant bit (LSB) of the node address. IP addresses are in decimal form therefore any reference made to the DA1 of the PLC Ethernet module in software configurations must be in decimals. To convert any hexadecimal number to decimal, the following example can be applied: [14]



In this project, the PLC is given the IP address 10.1.136.46. The router is for personal home network therefore the IP addresses associated with the devices in the network is classified under Class C.

The hardware configuration of this PLC Ethernet module unit is associated with the software configuration in CX-programmer, which will be further explained in Appendix A.

3.4 Variable Frequency Drive

OMRON SYSDRIVE 3G3MV – A2007 inverter is a variable frequency drive used in this project to alter the frequency of the electrical power supplied to the motor so in order to change the motor speed [5].

The SYSDRIVE 3G3MV-A2007 inverter is suitable for a variety of applications as it incorporates many convenient controls and I/O functions that are easy-to-use as well as open loop vector control function. The advantages of vector control function are that it ensures a torque output that is 150% of the rated motor torque at an output frequency of 1 Hz, allows powerful revolution at low frequencies and restrains the revolution fluctuation caused by the load [5].

Table 3-1 tabulated the list of function parameters of the 3G3MV inverter that need to be configured in order to control the 3 phase squirrel cage induction motor effectively.

Para-meter	Setting Range	Default Setting	Set Value	Name	Function of Current Setting	Usage
n001	0-9	1	4	Parameter write prohibit selection/parameter initialisation	The parameters within a range from n001 to n179 (i.e., function groups 1 through 4 settings) is set and monitored.	The purpose of enabling this function is to initialize parameters to default settings, prohibit parameters to be written, set parameters or change the monitor range of parameters.
n002	0,1	0	1	Control mode selection	Vector control mode (open loop)	The control mode for the Inverter is set to vector control mode.
n004	0-9	0	1	Frequency reference selection	Frequency reference 1 (n024) is enable	This function enables the input method of frequency references to be selected in remote mode.
n006	0,1	0	0	Reverse Rotation-prohibit selection	Reverse rotation is enabled.	Reverse rotation command sent to the Inverter is enabled.
n024	0 to maximum frequency	6.00	0.00	Frequency Reference 1	Internal frequency references are set.	Frequency references 1 through 8 are set and used for multi-step speed references.
n025		0.00	7.00	Frequency Reference 2		
n026		0.00	14.00	Frequency Reference 3		
n027		0.00	21.00	Frequency Reference 4		
n028		0.00	28.00	Frequency Reference 5		
n029		0.00	35.00	Frequency Reference 6		
n030		0.00	42.00	Frequency Reference 7		
n031		0.00	49.00	Frequency Reference 8		

n036	0%-150% of rated output current of the Inverter	Varies with the capacity	2.0 A	Rated motor current setting	The rated motor current which is used for the reference current of motor overload detection is set according to the rated motor current shown on the motor nameplate.	Rated motor current is set for use as a vector control constant and to determine the electronic thermal characteristics to protect the motor from overheating.
n050	1-25	1	1	Multi-function digital Input 1 (S1)	Forward/Stop function is enabled.	Forward and reverse rotation command in 2-wire sequence.
n051	1-25	2	2	Multi-function digital Input 2 (S2)	Reverse/Stop function is enabled.	
n054	1-25	6	6	Multi-function digital Input 5 (S5)	Multi-step speed reference 1	These signals are set to select frequency References 1 through 8.
n055	1-25	7	7	Multi-function digital Input 6 (S6)	Multi-step speed reference 2	
n056	1-25, 34 and 35	10	8	Multi-function digital Input 7 (S7)	Multi-step speed reference 3	
n106	0.0 to 20.0 (Hz)	Varies with the capacity	3.7 Hz (Refer Equation 3)	Rated motor slip setting	The rated slip value of the motor in use is set.	Rated motor slip set at 3.7Hz is used as vector control constant and slip compensation.
n110	0 to 99 (%)	Varies with the capacity	39% (Refer Equation 4)	Motor no-load current setting	The no-load current of the motor in use is set based on the rated motor current as 100%.	This parameter is set for used as vector control constant and slip compensation.

Table 3-1:List of Function Parameters and its Configuration in 3G3MV Inverter [5,26]

The rated motor slip value is calculated from the rated frequency (Hz) and rpm on the motor nameplate by using the following formula: [5,26]

Given that rated frequency = 50Hz; rated rpm = 1390; number of poles = 4,

$$S_{rated} = f_{rated} - N_{rated} \times \frac{P}{120} \quad \text{.....(Equation 3)}$$

$$= 50 - 1390 \times \frac{4}{120} = 3.667\%$$

Where

S_{rated} = Rated slip value
f_{rated} = Rated frequency (Hz)
N_{rated} = Rated rpm
P = Number of poles

The motor no-load current is set in percentage based on the rated current of the Inverter as 100%. [5,26]

With no load current at 50Hz = 0.779A,

$$Motor\ no\ load\ current = \frac{0.779}{2} \times 100 = 38.95\% \approx 39\% \quad \text{.....(Equation 4)}$$

4.0 PROJECT IMPLEMENTATION

4.1 Implementation Process

The Ethernet control systems presented in this thesis is to control squirrel cage three phase induction motor. In order to achieve the objectives of this project, the establishment of the communication between PLC and LabVIEW is crucial as LabVIEW is 3rd party software instead of using the software implemented in the PLC itself. Thus, the project used LabVIEW to perform the start and stop operation of the motor, either in forward or reverse direction, and varying the speed by changing the frequency of the motor. However, this project is not a supervisory control and data acquisition (SCADA) system since there is no practical data measurement acquire from the actual output of the motor.

This project has three layer network architecture illustrated in Figure 4.1.

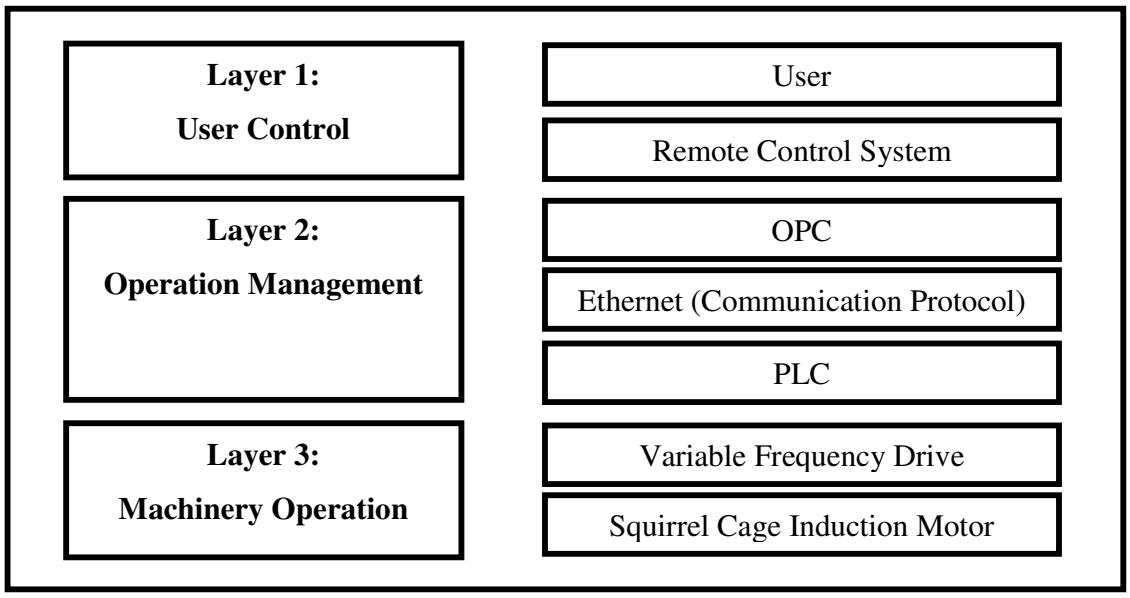


Figure 4-1: Three Layer Network Architecture

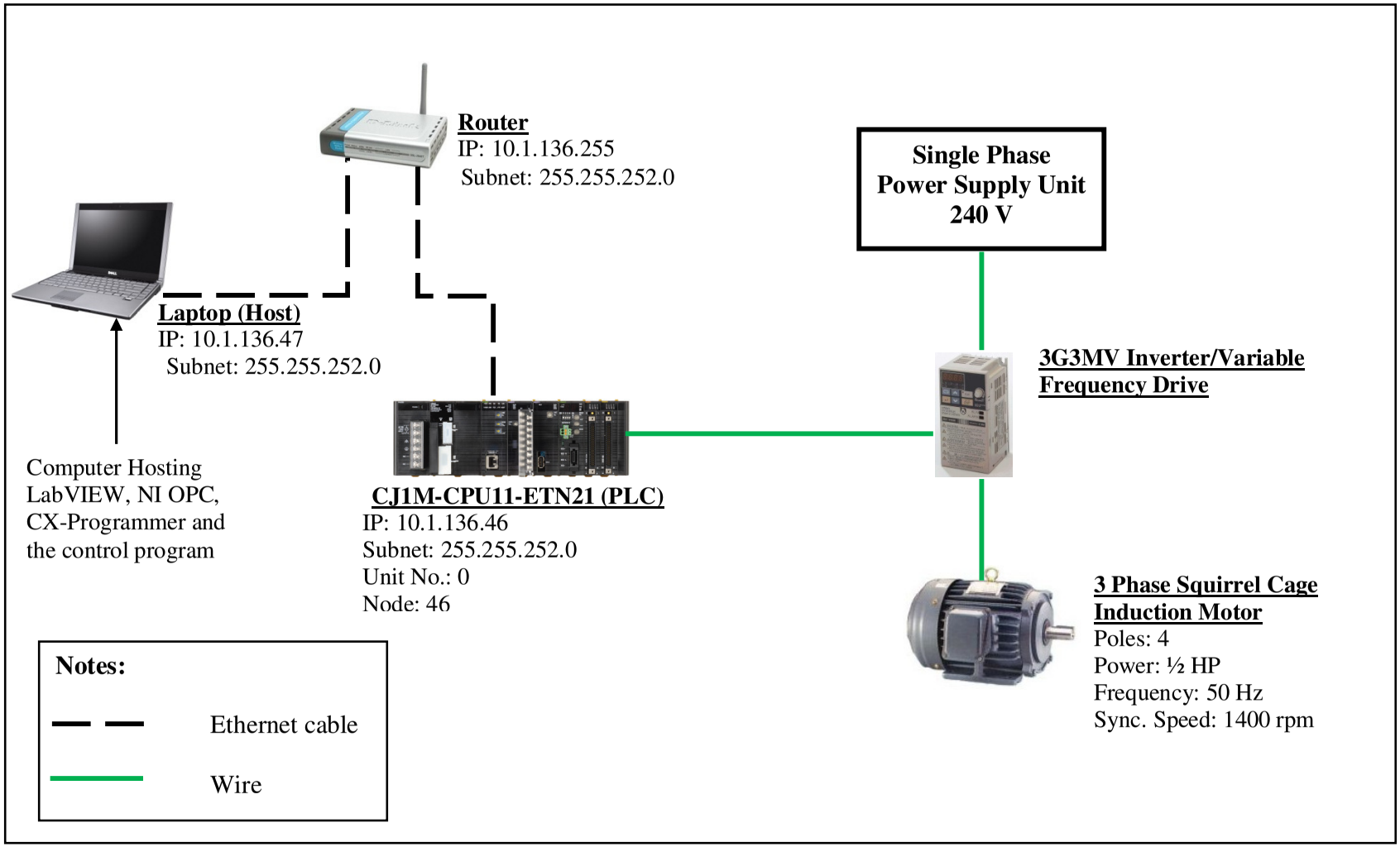


Figure 4-2: Actual Network Configuration

As illustrated in Figure 4-2, user gets the authority to control through the host computer, which is laptop in this thesis. Then the input data by the user will convert into Boolean data and send to CJ1M-CPU11-ETN21 Programmable Logic Controller through Ethernet cable and router. Once the Boolean data had been processed by the PLC, the relevant address in the basic output of the PLC will be turned on. This turn on process allow the 3G3MV Inverter / Variable Frequency Drive to operate the Three Phase Squirrel Cage Induction Motor according to the input data given by the user. In addition, 3G3MV Inverter / Variable Frequency Drive also serve as inverter between the power supply unit and the motor as the input power is single phase power, while the Squirrel Cage Induction Motor is operate in three phase power.

4.2 Implementation of VI Design

The objective of the VI program in this thesis is to allow user to make the decision of the start and stop operation of the motor, either in forward or reverse direction, and varying the speed by changing the frequency of the motor, by perform two simple step. Firstly select the turning direction by press the push button in the VI Front Panel, which is either forward or reverse. Secondly, vary the speed by turning the “frequency” knob to the value of the frequency that the user desire. Figure 4-3 illustrate the VI Front Panel.

Before running the VI, make sure all the hardware had been switched on and configured correctly, and launch the NI OPC Quick Client so that the OPC tags can be browsed by the shared variables in this VI.

There are 5 OPC tags had been created and used in this project, and the details of the tags had been tabulated in Table 4-1 below.

Table 4-1: The details of OPC tags and its connection to 3G3MV Inverter

OPC Tag name	PLC Address	3G3MV Inverter	
		Connected Terminal	Terminal Name
outputbit1	CIO0001.01	S1	Multi-function input 1 (Forward/Stop)
outputbit2	CIO0001.02	S2	Multi-function input 2 (Reverse/Stop)
outputbit 5	CIO0001.05	S5	Multi-function input 5 (Multi-step speed reference 1)
outputbit6	CIO0001.06	S6	Multi-function input 6 (Multi-step speed reference 2)
outputbit7	CIO0001.07	S7	Multi-function input 7 (Multi-step speed reference 3)

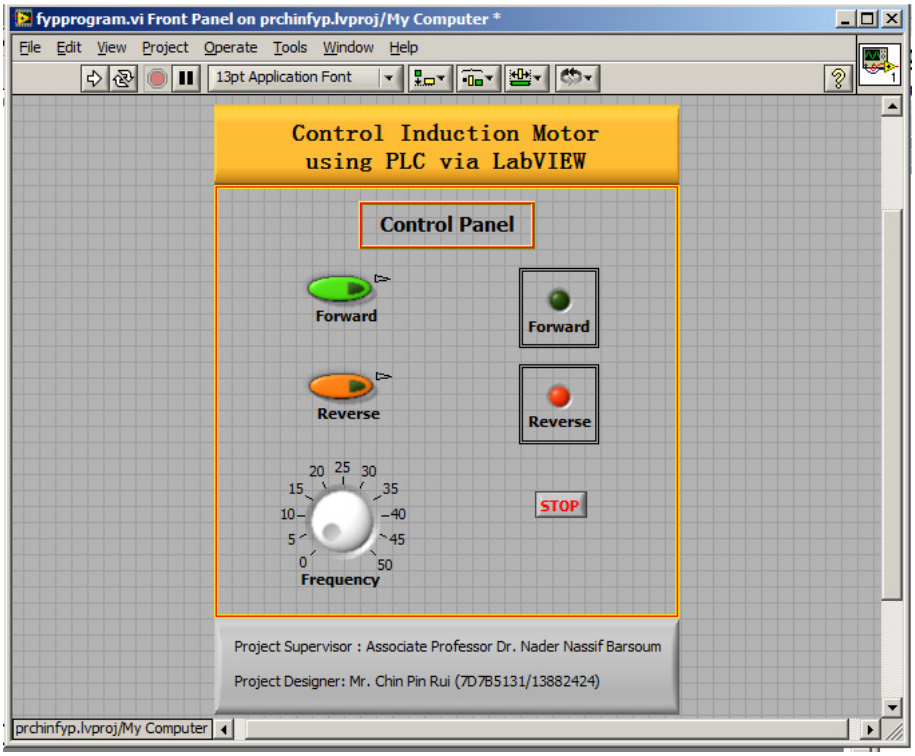


Figure 4-3: VI Front Panel of the project

By referring to Figure 4-3, the green push button is the switch to determine the motor is turning in forward direction. Meanwhile the orange push button is the switch to determine the motor is turning in reverse direction. Both green and orange light indicators at the right hand side shows whether the push button had been switch on. And then the knob labelled as "frequency" is the key program to control the frequency as well as the speed of the motor. The push button labelled "stop" function to stop the program execution.

Figure 4-4 illustrate the VI Block Diagram, which the programming part of the VI.

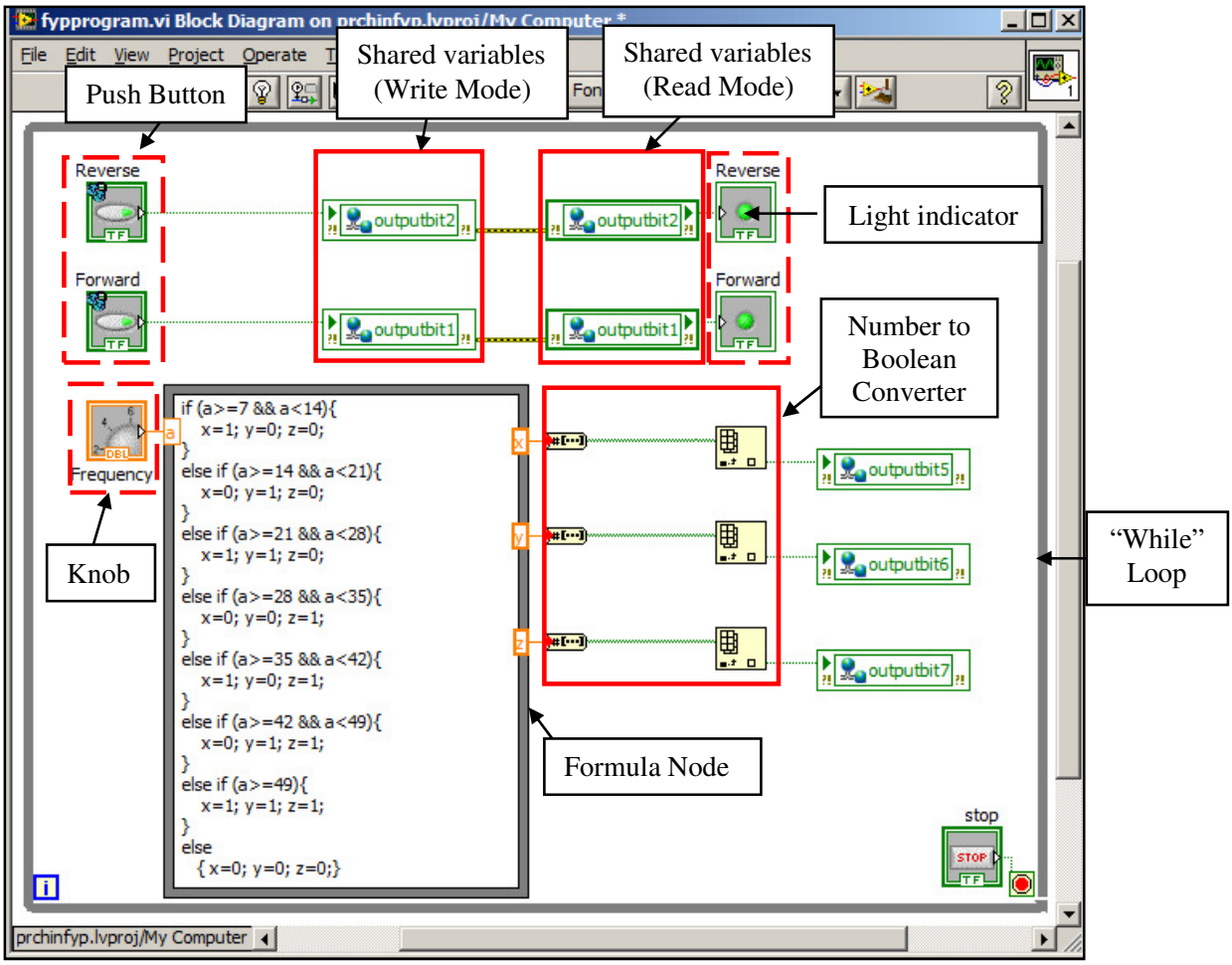


Figure 4-4: VI Block Diagram

By referring to the Figure 4-4, the red colour square box with dot lines are the VI components that are visible in both Front Panel and Block Diagram. Example the push buttons, knob, light indicator. Meanwhile the red colour square box without dot lines are the VI component that is visible in Block Diagram but they are not visible in Front Panel, which are essential VI to structure the program. For example, shared variables, formula node, number to boolean converter (consist of number to boolean array and index array) and the While loop. The While loop, which is similar concept with the While loop in C Programming, is used in this VI to ensure the program execute continuously until the stop button had been triggered.

The VI block diagram consist of two parts of program. First part of the program is to allow user to switch on either Forward or Reverse direction of the squirrel cage induction motor. And the second part of the program is to vary the frequency of the motor by changing the knob value.

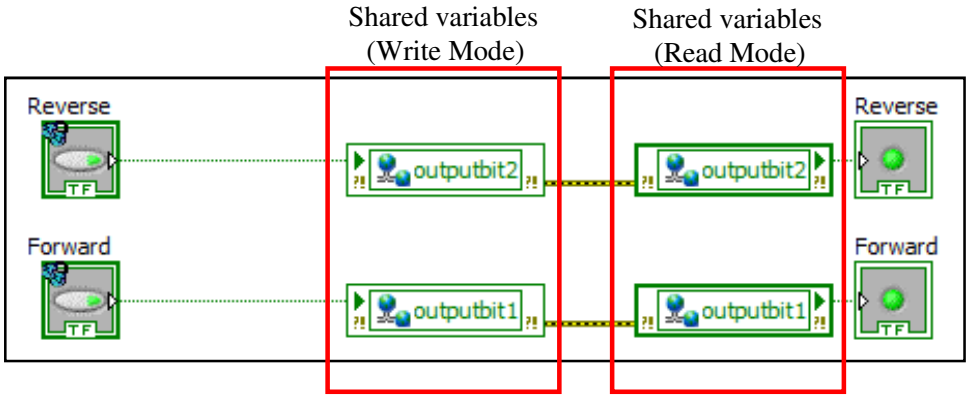


Figure 4-5: VI Block Diagram Program (Motor Orientation Switching Part)

By referring to Figure 4-5, if user had switch on the Forward button, the signal will be transmitted to the “outputbit1” shared variable, which is in write mode, i.e. the signal will

be write into the PLC. Then once the signal had been successfully write to the PLC, the output address 1.01 of the PLC will light up, and as well as the "Forward" light indicator in the LabVIEW Front Panel. This process identical to the reverse direction as well.

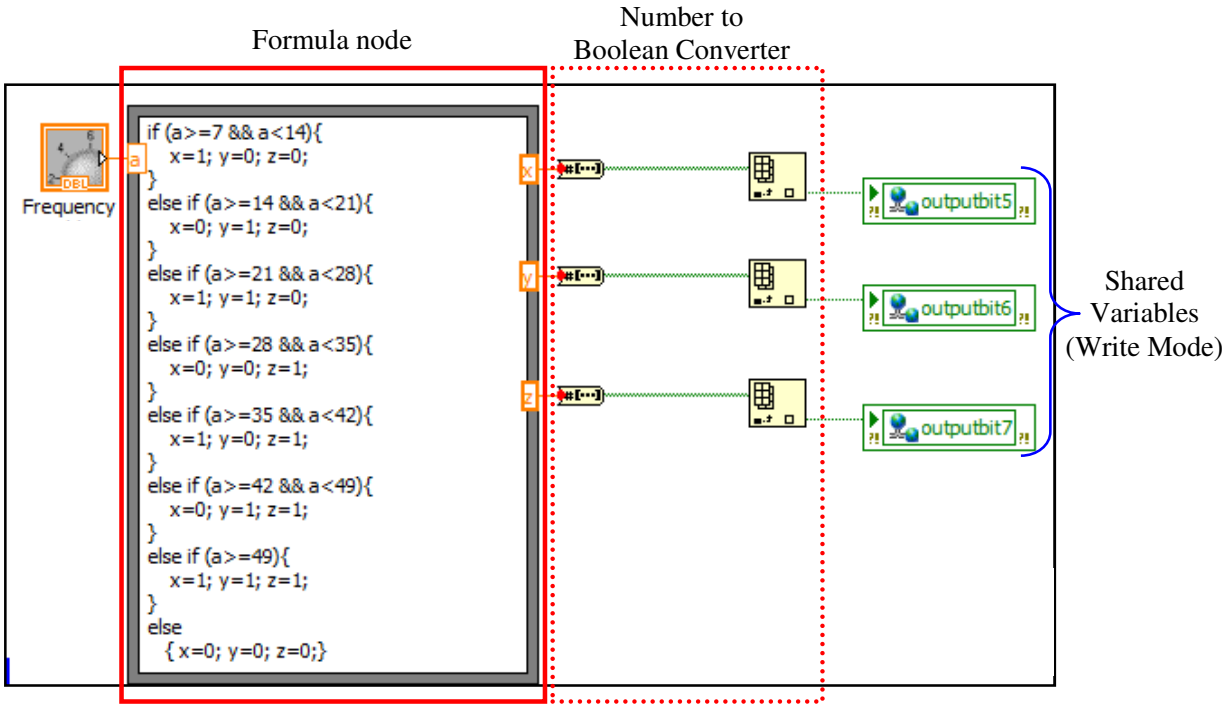


Figure 4-6: VI Block Diagram Program (Motor Frequency Varying Part)

By referring to the Figure 4-6, the red colour square box indicate the VI named formula node, where the language used in the formula node is C programming. after the user set the value of the frequency by turn the "Frequency Variable" knob, the value will send to the formula node, where it is denoted as "*a*". The explanation of the C programming language inside the formula node is shown in Table 4-2.

The output of the formula node had been divided into x, y and z, in which the number is either 0 or 1. In order to be readable by the shared variables, it is necessary to convert the number into boolean format. And once the signal is in boolean format and send to the

shared variable, the information will be write into the PLC, and the output address 1.05, 1.06, 1.07 of the PLC will light up according to the desired output.

Table 4-2: VI Formula Node C Programming and its Explanation

C Programming Code	Explanation

if (a>=7 && a<14){
x=1; y=0; z=0;
}
	If *a* is more than or equal to 7 AND less than 14, then x equals to 1, y equals to 0 and z equals to 0

else if (a>=14 && a<21){
x=0; y=1; z=0;
}
	If *a* is more than or equal to 14 AND less than 21, then x equals to 0, y equals to 1 and z equals to 0

else if (a>=21 && a<28){
x=1; y=1; z=0;
}
	If *a* is more than or equal to 21 AND less than 28, then x equals to 1, y equals to 1 and z equals to 0

else if (a>=28 && a<35){
x=0; y=0; z=1;
}
	If *a* is more than or equal to 28 AND less than 35, then x equals to 0, y equals to 0 and z equals to 1

else if (a>=35 && a<42){
x=1; y=0; z=1;
}
	If *a* is more than or equal to 35 AND less than 42, then x equals to 1, y equals to 0 and z equals to 1

else if (a>=42 && a<49){
x=0; y=1; z=1;
}
	If *a* is more than or equal to 42 AND less than 49, then x equals to 0, y equals to 1 and z equals to 1

else if (a>=49){
x=1; y=1; z=1;
}
	Else if *a* is more than or equal to 49, then x equals to 1, y equals to 1 and z equals to 1

else { x=0; y=0; z=0;}	Or else x equals to 0, y equals to 0 and z equals to 0

Table 4-3 below explained the relationship between the multi-step speed references 1 through 3 and frequency references 1 through 8.

Table 4-3 Relationship between multi-step speed references and frequency references [5]

Frequency reference	Multi-step speed reference 1 (Set value: 6)	Multi-step speed reference 2 (Set value: 7)	Multi-step speed reference 3 (Set value: 8)
Frequency reference 1	OFF	OFF	OFF
Frequency reference 2	**ON**	OFF	OFF
Frequency reference 3	OFF	**ON**	OFF
Frequency reference 4	**ON**	**ON**	OFF
Frequency reference 5	OFF	OFF	**ON**
Frequency reference 6	**ON**	OFF	**ON**
Frequency reference 7	OFF	**ON**	**ON**
Frequency reference 8	**ON**	**ON**	**ON**

The concept of frequency varying in Table 4-3 is crucial as the PLC communicate with the variable frequency drive in Boolean format. The value of each frequency reference can be set in the function parameters of the variable frequency drive (refer to Table 3-1). In this thesis, the value of each frequency reference had been tabulated in both Table 3-1 and Table 4-4.

Table 4-4: Conditions of Input Frequency in LabVIEW for varying frequency reference

Frequency Input Range in LabVIEW (Hz)	Multi-step speed reference 1 (Outputbit5)	Multi-step speed reference 2 (Outputbit6)	Multi-step speed reference 3 (Outputbit7)	Output Frequency of Squirrel Cage Induction Motor (Hz)
$x < 7$	0	0	0	0
$7 \leq x < 14$	1	0	0	7
$14 \leq x < 21$	0	1	0	14
$21 \leq x < 28$	1	1	0	21
$28 \leq x < 35$	0	0	1	28
$35 \leq x < 42$	1	0	1	35
$42 \leq x < 48$	0	1	1	42
$x > 48$	1	1	1	49

By referring to Table 4-4, as long as the user had set the input frequency according to any of the condition, the number will be converting into Boolean in which it will arrange according to the relevant Multi-step speed reference. Then this Boolean number will trigger the variable frequency drive and determine the frequency and as well as the speed of the squirrel cage induction motor according to the output condition listed in Table 4-4. For example, if the user set the input as 8 Hz, in which the input range of this value is within $7 \leq x < 14$. Therefore the Boolean number of Multi-step speed reference 1 is 1, and the rest are 0. And this signal triggers the variable frequency drive to deliver the frequency of 7 Hz.

5.0 Testing and Verification

After the implementation, test is performed in order to verify that the project is working smoothly.

Before implement the test process, it is important to refer and follow the checklist tabulated in Table 5-1. If problem(s) arise during the execution of the project, revise the checklist to troubleshoot the problem(s) and identify the suitable solution(s). Figure 5-1 and Figure 5-2 shows the overview of the project.

Table 5-1: Checklist before project testing

No.	Task	Notes
1.	Make sure all the hardware had been switched on. - User Computer - Ethernet Router - PLC - VFD	-
2.	Make sure all the wiring of the hardware had been connected properly. - LAN cables (PC to PLC) - Wires (PLC to resistors) - Wires (resistors to VFD)	
3.	Make sure the hardware of the PLC had been configured correctly.	Refer to Page 38 for details
4.	Make sure the software of the PLC had been configured correctly.	Refer to Appendix A for details
5.	Make sure the NI OPC Servers had been setup and configured correctly.	Refer to Appendix B for details
6.	If No. 5 had been done, launch the NI OPC Client.	Refer to Appendix B for details
7.	Make sure the LabVIEW shared variables are configured correctly.	Refer to Appendix C for details

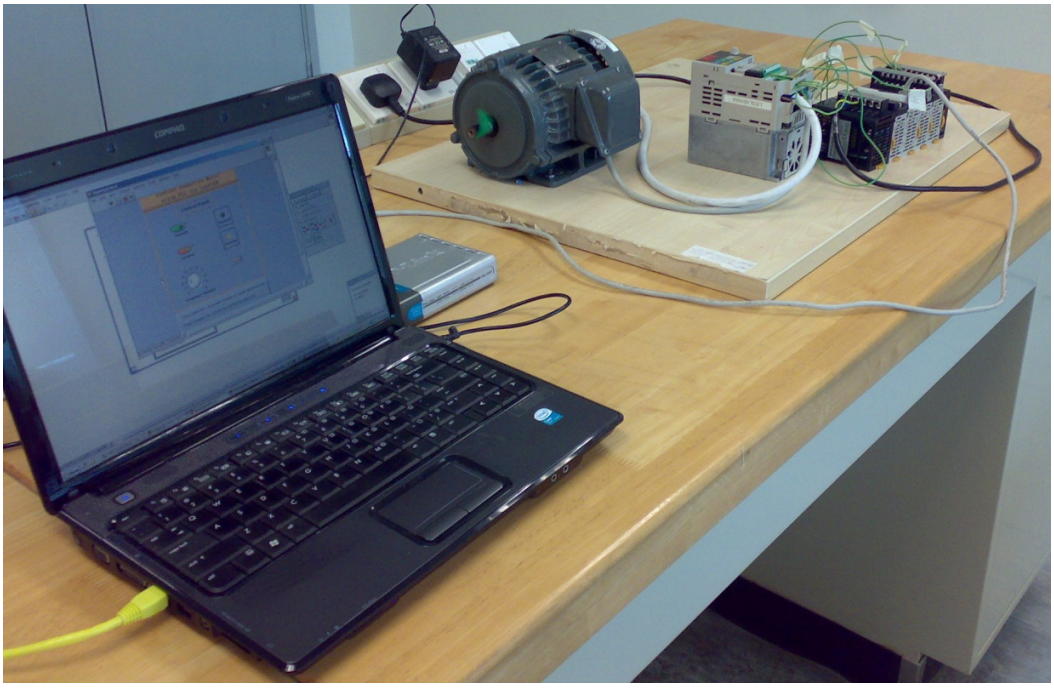


Figure 5-1: Side view of the project

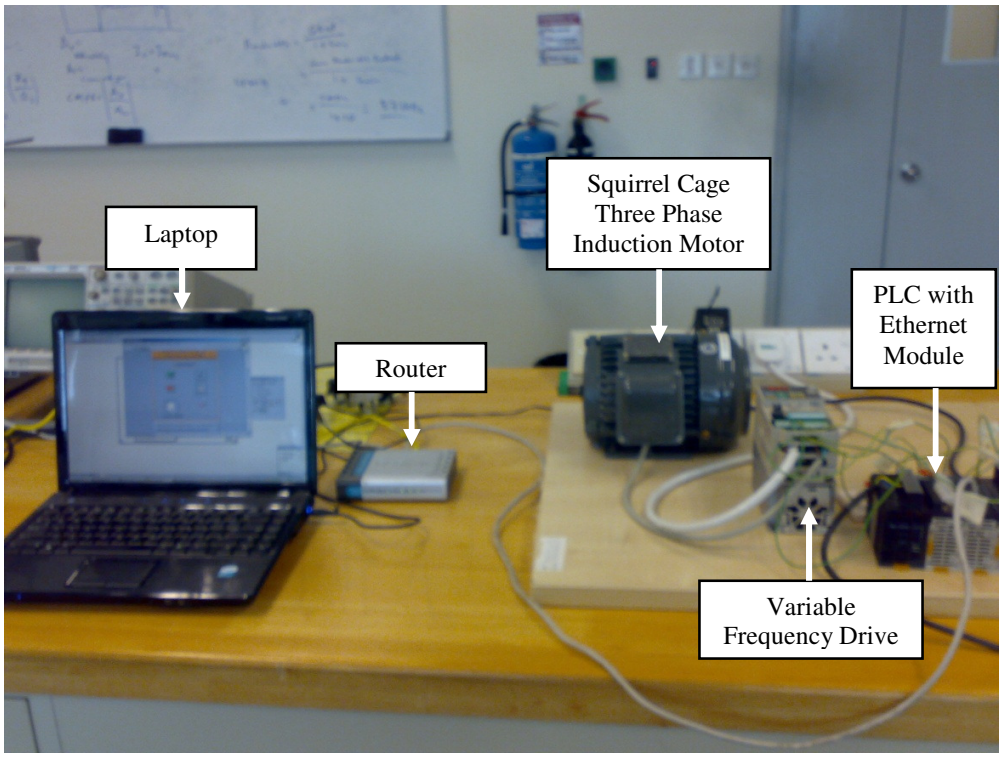


Figure 5-2: Front view of the project

If all the tasks in Table 5-1 had been performed correctly, run the project VI in LabVIEW. Figure 5-3 shows the PLC Ethernet Module light indicators that light up when the project VI is executing. And Figure 5-4 shows the snapshot of the Squirrel Cage Three Phase Induction Motor when it is running.

Figure 5-3: PLC Ethernet Module Light Indicator Status

Figure 5-4: Squirrel Cage Three Phase Induction Motor running in forward direction

If the user set the orientation of the induction motor into forward direction, the outcome that can be seen in the PLC will be identical with the situation in Figure 5-5. And if the input frequency set by the user is less than 7 Hz, the VFD frequency meter will indicate 0 Hz and the motor will remain idle, in Figure 5-6..

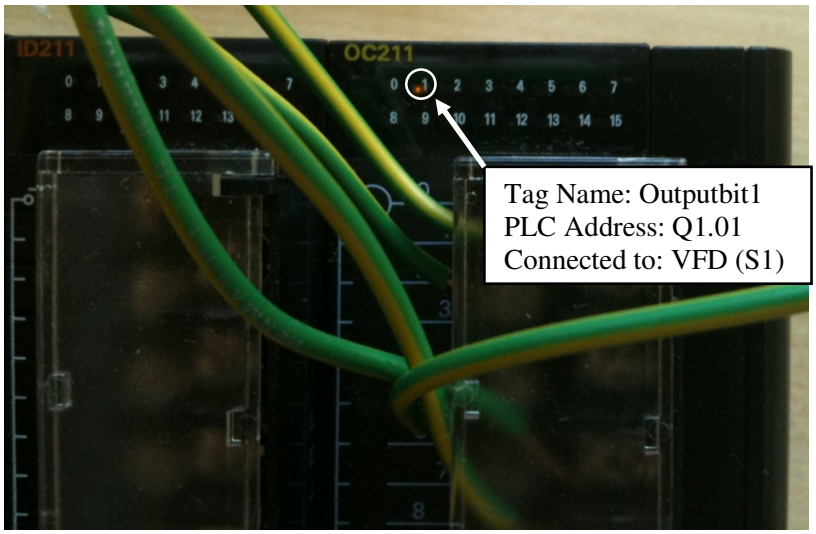


Figure 5-5: PLC Output for forward direction and frequency = 0 Hz

Figure 5-6: VFD frequency meter for forward direction and frequency = 0 Hz

Meanwhile if the user set the orientation of the induction motor into forward direction, the outcome that can be seen in the PLC will be identical with the situation in Figure 5-7. Similarly if the input frequency set by the user is less than 7 Hz, the VFD frequency meter will indicate 0 Hz and the motor will remain idle, in Figure 5-8.

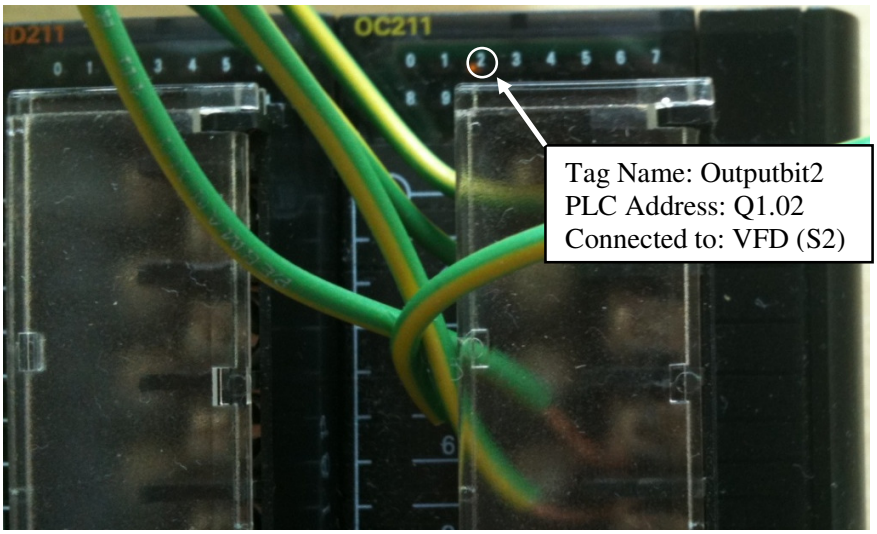


Figure 5-7: PLC Output for reverse direction and frequency = 0 Hz

Figure 5-8: VFD frequency meter for reverse direction and frequency = 0 Hz

When user decided to set the coast of the induction motor in forward direction, and the input frequency range is from 7 Hz until less than 14 Hz, the outcome that can be seen in the PLC and the VFD frequency meter will be identical with the situation in Figure 5-9 and Figure 5-10 respectively.

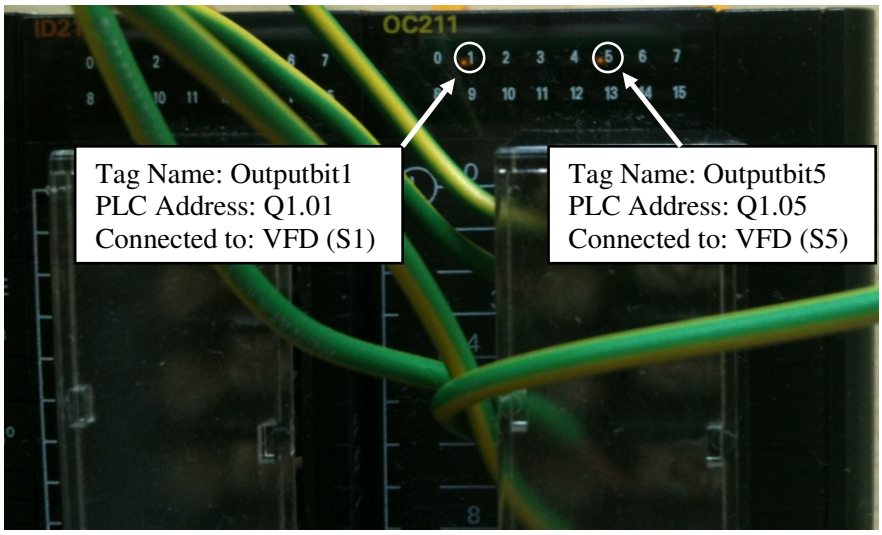


Figure 5-9: PLC Output for forward direction and frequency = 7 Hz

Figure 5-10: VFD frequency meter for forward direction and frequency = 7 Hz

If the user increases the input frequency range to a value that is within 14 Hz and less than 21 Hz, then the outcome that can be seen in the PLC and the VFD frequency meter will be identical with the situation in Figure 5-11 and Figure 5-12 respectively.

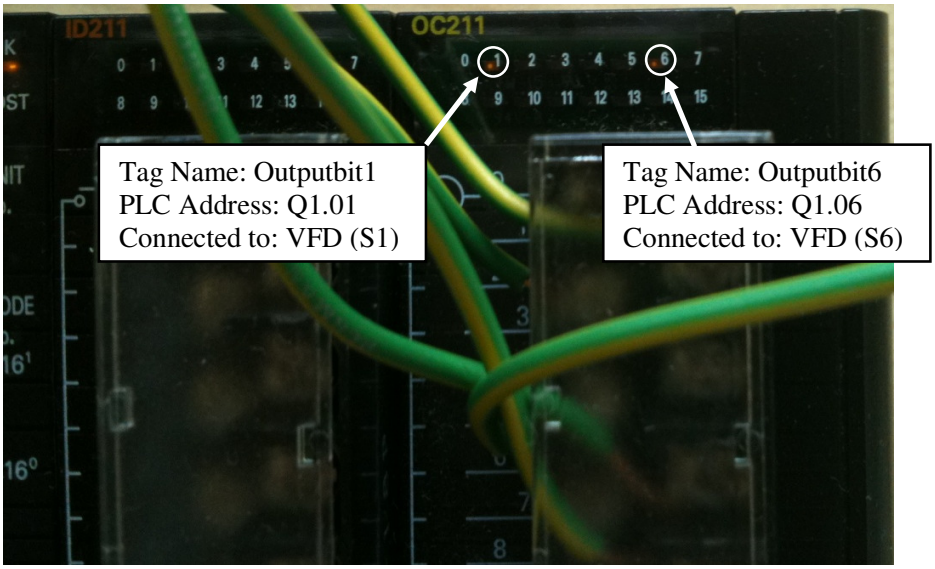


Figure 5-11: PLC Output for forward direction and frequency = 14 Hz

Figure 5-12: VFD frequency meter for forward direction and frequency = 14 Hz

If the user increases the input frequency range to a value that is within 21 Hz and less than 28 Hz, then the outcome that can be seen in the PLC and the VFD frequency meter will be identical with the situation in Figure 5-13 and Figure 5-14 respectively.

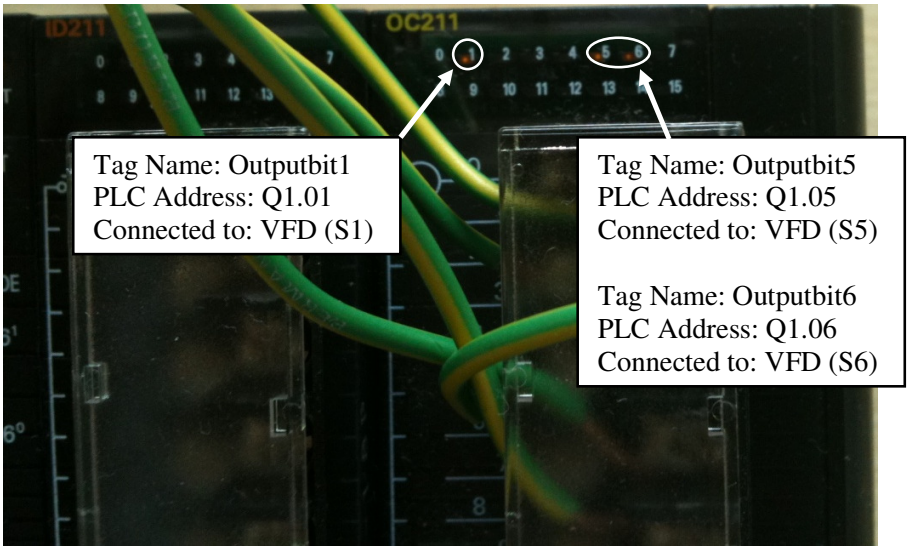


Figure 5-13: PLC Output for forward direction and frequency = 21 Hz

Figure 5-14: VFD frequency meter for forward direction and frequency = 21 Hz

If the user increases the input frequency range to a value that is within 28 Hz and less than 35 Hz, then the outcome that can be seen in the PLC and the VFD frequency meter will be identical with the situation in Figure 5-15 and Figure 5-16 respectively.

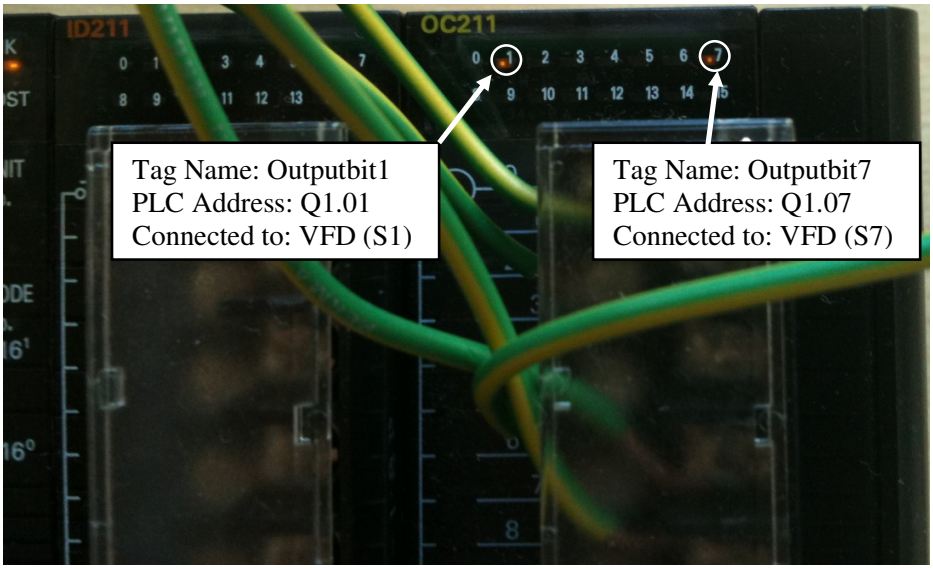


Figure 5-15: PLC Output for forward direction and frequency = 28 Hz

Figure 5-16: VFD frequency meter for forward direction and frequency = 28 Hz

If the user increases the input frequency range to a value that is within 35 Hz and less than 42 Hz, then the outcome that can be seen in the PLC and the VFD frequency meter will be identical with the situation in Figure 5-17 and Figure 5-18 respectively.

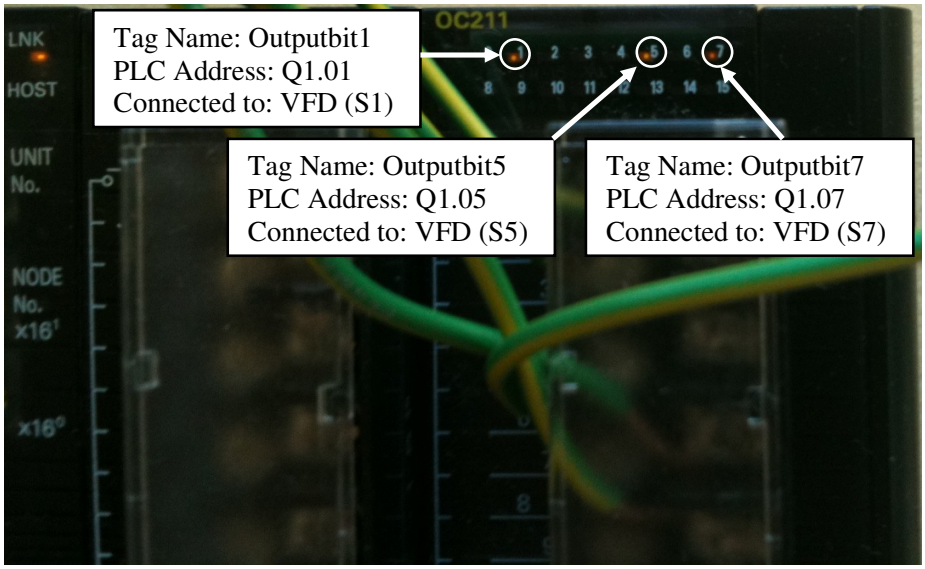


Figure 5-17: PLC Output for forward direction and frequency = 35 Hz

Figure 5-18: VFD frequency meter for forward direction and frequency = 35 Hz

If the user increases the input frequency range to a value that is within 42 Hz and less than 49 Hz, then the outcome that can be seen in the PLC and the VFD frequency meter will be identical with the situation in Figure 5-19 and Figure 5-20 respectively.

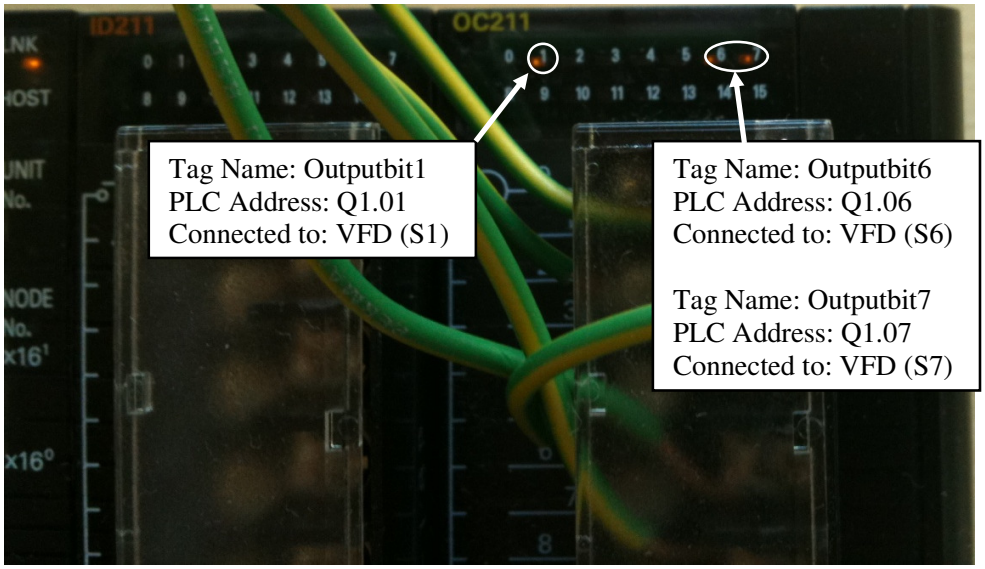


Figure 5-19: PLC Output for forward direction and frequency = 42 Hz

Figure 5-20: VFD frequency meter for forward direction and frequency = 42 Hz

If the user increases the input frequency range to a value that is equal or more than 49 Hz, then the outcome that can be seen in the PLC and the VFD frequency meter will be identical with the situation in Figure 5-21 and Figure 5-22 respectively.

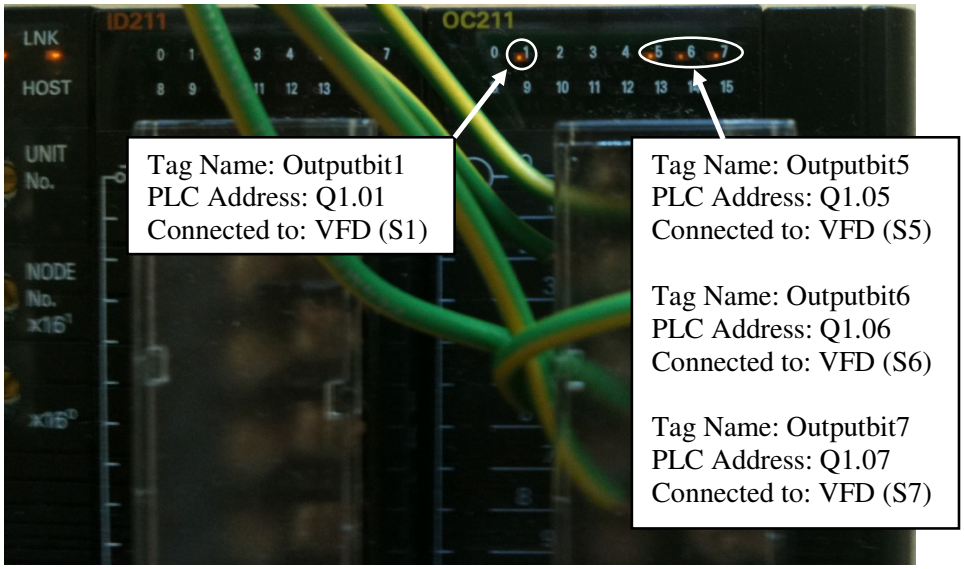


Figure 5-21: PLC Output for forward direction and frequency = 49 Hz

Figure 5-22: VFD frequency meter for forward direction and frequency = 49 Hz

6.0 Conclusion & Recommendation

6.1 Conclusion

In conclusion, the objective and the scope of the project had been achieved. All the basic requirements of the projects had been achieved. But there are imperfections in this project. Firstly there is a delay of time during the project execution, which every decision made by the user through LabVIEW will takes about 30 seconds in average to send the signal to the PLC. The delay mainly cause by the poor specification of the laptop in this project, that lead the LabVIEW takes long time to execute the program and generate the Boolean data to send to the PLC.

Secondly, the Ethernet connectivity between the laptop and the PLC is easily disconnected. This problem arises also due to the poor specification of the laptop used in this project. This problem happen when the laptop had used up too much memory, causing the laptop lag and unable to use temporary. Therefore user cannot make too many actions and changes at the same time as it will crash the program. Thirdly, there is limitation in the speed control as there are only 3 multi-step references in the VFD. The frequency changes in this project cannot perform in smaller steps unless there are additional multi-step references provided.

But in overall, choosing LabVIEW as the human machine interface of this project is a correct decision as it has various types of applications and functions that are easy to understand and use. Additionally, this approach is more economical as the objectives of

the project had achieved with only basic functionality of the LabVIEW toolkit used, which are shared variables and NI OPC Servers.

OMRON CJ1 series PLC is easy to install and setup. Both hardware and software configuration can be easily done. It can carry out additional functions by simply add more units with various functions, which in this project is the Ethernet unit.

The 3G3MV inverter is a user friendly VFD that allows the user to configure the function parameter easily and the circuit wiring can be completed easily.

6.2 Recommendation

Ethernet control of the squirrel cage induction motor is not consider as a perfect distance remote control system as it is not a supervisory control and data acquisition (SCADA) system. It is crucial to acquire data from the induction motor so that user can make decision according to the current status of the motor. For example, if the data acquired from the sensor indicates that the temperature of the motor is too high, user can immediately get warning from the system and stop the motor from running. LabVIEW has the data acquisition (NI DAQ) toolkit that allow the acquire data from the motor to be processed and become useful signal that is readable by the user. Additionally, CJ1 series PLC has the MAD 42 unit that allow the analogue signal from the sensor to be process by the PLC by simply connect the relevant sensors to the MAD 42 unit.

Furthermore, by using LabVIEW internet toolkit, the remote control system can be upgraded into web based control. Other computers can have the access to control of the VI the system provided they had installed the NI Runtime (RT) engine plug-in that has similar version with the main user computer. Microsoft Silver light plug-in is also required to view the embedded VI java-script on the web.

7.0 Bibliography

[1] PLC Manual. Introduction to PLCs. [Online]. http://www.plcmanual.com/

[2] Suffolk Automation. (2005, December) Basic of Programmable Logic Controller. [Online]. http://www.suffolk-automation.co.uk/uploads/pdfs/LOGIC_CONTROLLERS.pdf

[3] PLC Manual. PLC History. [Online]. http://www.plcmanual.com/plc-history

[4] Phil Melore. PLC History. [Online]. http://www.plcs.net/chapters/history2.htm

[5] Nader Nassif Barsoum, "PLC Humidity Control Inverter Fed Induction Motor," Curtin University Sarawak Campus, Miri, Undergraduate Thesis 2009.

[6] Clive Maynard. PLC Structure. [Online]. http://kernow.curtin.edu.au/www/plc/2STRUCT.HTM

[7] Siemens. Basic PLC Operation. [Online]. http://sea.siemens.com/step/templates/lesson.mason?plcs:1:2:2

[8] Joliet Technologies. What is a Variable Frequency Drive? [Online]. http://www.joliettech.com/what_is_a_variable_frequency_drive.htm

[9] Yarn Spinning Technology. Induction Motor. [Online]. http://textiletechnology.bravehost.com/spinning/inductionmotor.htm

[10] Nader Nassif Barsoum. (2007, April) AC Motor. [Online]. http://www.engineeringedu.org/cources/units/es/2a.pdf

[11] Halit Eren, Instrumentation Network, Curtin University Instrumentation and Control 402 Lecture Notes 4-6.

[12] University of South Florida. (2009) Chapter 2: Protocol. [Online]. http://fcit.usf.edu/network/chap2/chap2.htm

[13] CISCO. (2010, April) CISCO. [Online]. http://www.cisco.com/web/strategy/docs/manufacturing/industrial_ethernet.pdf

[14] Nader Nassif Barsoum, "Ethernet LabVIEW Control," Curtin University Sarawak Campus, Miri, Undergraduate Thesis 2010.

[15] Search Networking.com. (2000, August) What is Ethernet? [Online]. http://searchnetworking.techtarget.com/definition/Ethernet

[16] Javvin Network Management and Security. Ethernet: IEEE 802.3 Local Area Network (LAN) protocols. [Online]. http://www.javvin.com/protocolEthernet.html

[17] University of South Florida. (2009) Chapter 5: Topology. [Online]. http://fcit.usf.edu/network/chap5/chap5.htm

[18] How Stuff Works. (2001, January) What is an IP Address? [Online]. http://computer.howstuffworks.com/internet/basics/question549.htm

[19] National Instruments. What is OPC? [Online]. http://zone.ni.com/devzone/cda/tut/p/id/7451

[20] National Instruments. LabVIEW FAQs. [Online]. http://www.ni.com/labview/faq.htm

[21] National Instruments, "Getting Started with LabVIEW," National Instruments, 373427F-01, 2009.

[22] National Instruments. Connect LabVIEW to Any PLC Using OPC. [Online]. http://zone.ni.com/devzone/cda/tut/p/id/7450

[23] National Instruments. Using the LabVIEW Shared Variable. [Online]. http://zone.ni.com/devzone/cda/tut/p/id/4679

[24] OMRON Industrial Automation, "SYSMAC CJ Series Programmable Controllers," Operation Manual W393-E1-14, 2009.

[25] OMRON Industrial Automation, "CJ1M CPU Units with Ethernet Functions," Operational Manual 2005.

[26] OMRON Industrial Automation, "Sysdrive 3G3MV Multi-function Compact Inverter ," User Manual 2001. [Online]. http://www.limasoft.eu/omron/pdf/3G3MV_en_manual.pdf

[27] National Instruments. Basic TCP/IP Communication in LabVIEW. [Online]. http://zone.ni.com/devzone/cda/tut/p/id/2710

[28] National Instruments. Connect LabVIEW to Any Industrial Network and PLC. [Online]. http://zone.ni.com/devzone/cda/tut/p/id/5407

[29] National Instruments. Controlling a 3-Phase A/C Motor Using LabVIEW and FieldPoint. [Online]. http://zone.ni.com/devzone/cda/tut/p/id/4002

[30] CISCO. (2010) Ethernet - A Brief History. [Online]. http://www.cisco.com/en/US/docs/internetworking/technology/handbook/Ethernet.html#wp1020560

[31] Wikipedia. (2010, September) Industrial Ethernet. [Online]. http://en.wikipedia.org/wiki/Industrial_Ethernet

[32] Wu Y.C., Chang W.F., Chiu C.W., and Yu W.C. (2006, October) Feng Chia University. [Online]. http://dspace.lib.fcu.edu.tw/bitstream/2377/2285/1/ce07ics002002000276.PDF

[33] National Instruments. Connecting LabVIEW to an OPC Server through a DataSocket Connection. [Online]. http://zone.ni.com/devzone/cda/tut/p/id/3978

[34] National Instruments. DataSocket Tutorial. [Online]. http://zone.ni.com/devzone/cda/tut/p/id/3224

[35] National Instruments. DataSocket Simplifies Live Data Transfer for LabVIEW. [Online]. http://www.ni.com/pdf/datasocket/us/datasocketarticle.pdf

[36] National Instruments. Integrating the Internet into Your Measurement System (Datasocket Technical Overview). [Online]. http://www.ni.com/pdf/wp/wp1680.pdf

[37] Nader Nassif Barsoum, "PLC Inverter Fed Induction Motor," Curtin University Sarawak Campus, Miri, Undergraduate Thesis 2007.

[38] Omron. (2001) Sysdrive 3G3MV Multi-function Compact Inverter User Manual. [Online]. http://www.limasoft.eu/omron/pdf/3G3MV_en_manual.pdf

[39] TECO. Three Phase Squirrel Cage Induction Motor Catalogue. [Online]. http://www.teco.co.th/admin2/images/e-catalog/Catalogue%20AEEB.pdf

[40] Solomon S. Turkel. (1999, April) Understanding Variable Speed Drives (Part 2). [Online]. http://ecmweb.com/mag/electric_understanding_variable_speed_3/index.html

[41] University of Koblenz. (1999, April) Chapter 7: Ethernet/IEEE 802.3. [Online]. http://www.uni-koblenz.de/~ros/Rechnerorganisation/ethernet.pdf

8.0 Appendix

Appendix A: Ethernet Configuration in CX-Programmer

In order to use the Ethernet Module of the OMRON CJ1M-CPU11 PLC, the configuration needs to be done through CX-programmer in order to set the unit number and IP address of the PLC. Before setting up the configuration, user has to make sure that the correct version of CX-Programmer which is compatible with the PLC had been installed properly in the computer. The demonstration in this section is performing under the Windows Vista and Windows XP operating system. Therefore the explanation might be slightly different if the users are using other operating system in their computer.

Before started, it is important that the PLC had been turned on and identify the correct serial communication port in the computer. For laptop, additional hardware might be is needed, which is FTDI USB-Serial Port Converter Cable. In Windows Vista and Windows XP, left click **Start**, and then select and right click **My Computer >> Properties**, and then the System Properties window will appear.

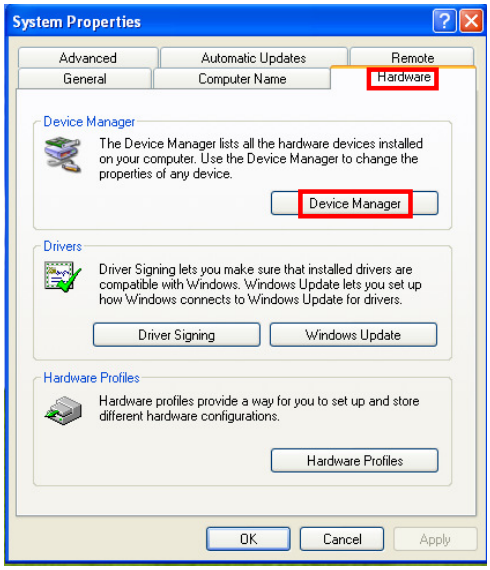


Figure 8-1: System Properties window in Windows XP

For Windows XP, select the **Hardware** tab, and left click the **Device Manager**. Meanwhile for Windows Vista, look at the top right of the window, under the **Task**, left click the **Device Manager,** Figure 8-2.

Afterwards, expand the **Ports (COM & LPT)**, and select the relevant port. The actual port is used in this project COM35, however the COM1 will be used as demonstration purpose only, Figure 8-2.

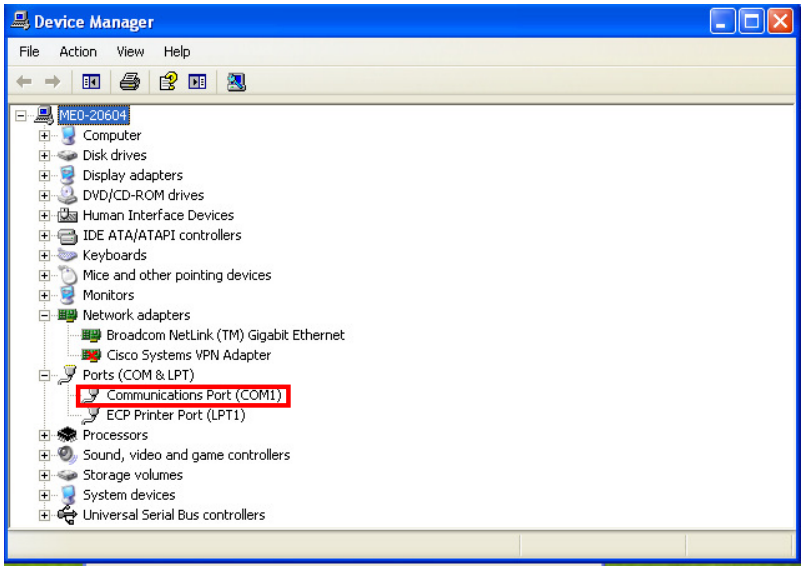


Figure 8-3: Device Manager window in Windows XP

Double click the COM1 and the properties dialog will appear. Left click the **Port Settings** tab and make configuration as demonstrated below. The configuration in **Port Settings** must follow and identical with the CX-programmer **Network Settings [SYSMAC WAY]** dialog, shown in Figure 8-3.

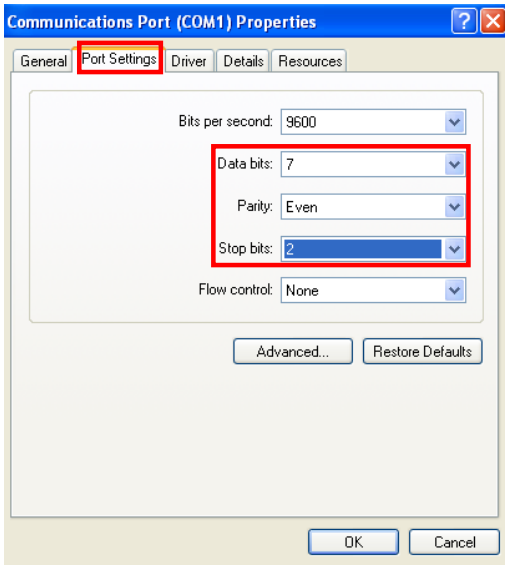


Figure 8-4: Communication Port Properties window and its configuration

Firstly, launch the CX-Programmer by left click the **Start >> All Programs >> OMRON >> CX-One >> CX-Programmer >> CX-Programmer**. Then left click the **File >> New** or press **Ctrl + N** to open a new blank document, and the **Change PLC** dialog will appear. In this dialog, change the "*Network Type*" into SYSMAC WAY, as illustrated in Figure 8-4.

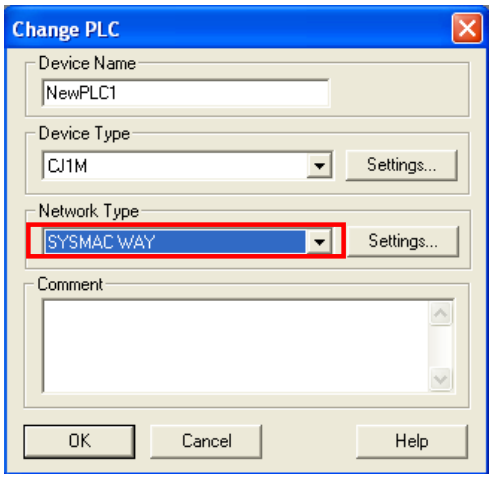


Figure 8-5: Change PLC dialog in CX-Programmer (SYSMAC WAY)

Next, left click the **Settings** button which is right hand side of the "*Network Type*", and the **Network Settings [SYSMAC WAY]** dialog will appear. Left click the **Driver** tab, and under the "*Connection*" category, change the "*Port Name*" to the suitable port, and which in this demonstration is COM 1. Make sure the Data Format shown in this dialog is identical with the port setting in Figure 8-5.

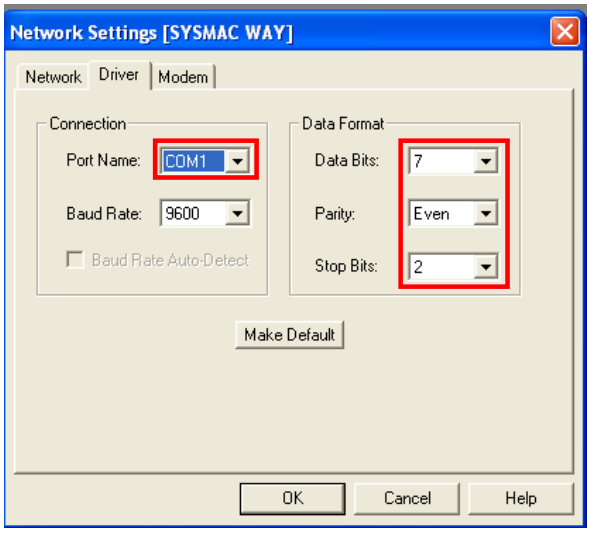


Figure 8-6: Network Settings of SYSMAC WAY in CX-programmer

Once all the previous steps had been correctly done, go back to the CX-Programmer and left click the online icon,  to connect the PC with the PLC. And then the Connect to PLC notification dialog will pop up. Left click **Yes** button to continue Figure 8-6.

Once the computer is connected to the PLC, configuration can be done much easier by changing from **Running Mode** into **Program Mode**. So, right click the **NewPLC1 [CJ1M] Running Mode**, which is located at the project window. Then select **Operating Mode >> Program**.

After that, double click the **IO Table and Unit Setup** in the project window, then the **PLC IO Table** will pop up. Left click the **Option** tab and select **Create** to create the I/O Table automatically. And then perform the actions as demonstrated in Figure 8-6.

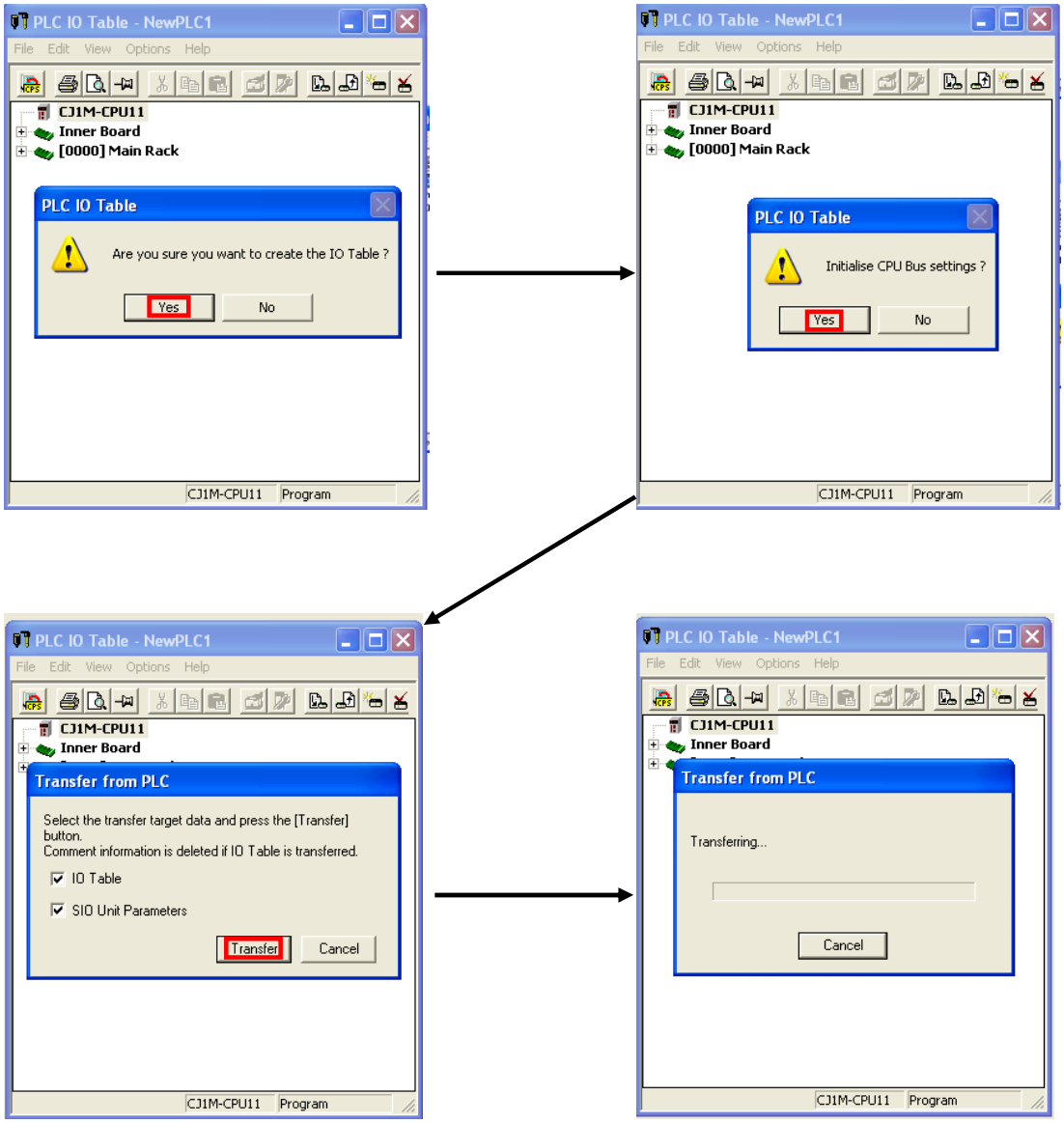


Figure 8-7: Demonstration of Create PLC IO Table in CX-Programmer

The status of the data transfer from PLC to computer will be shown when the **Transfer Result** dialog shows up. If it is successful the dialog will write “Transfer Success: 1 Unit” and “Transfer Unsuccessful: 0 Unit”. Press **OK** button to close the dialog. Meanwhile in **PLC IO Table** dialog, under the **[0000] Main Rack**, the relevant unit of the PLC are all set, as shown in the Figure 8-7.

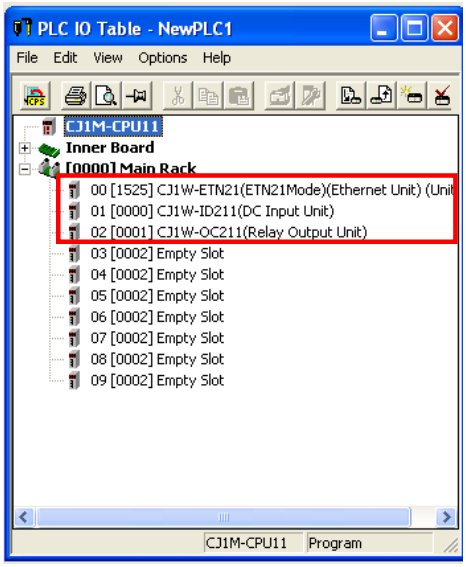


Figure 8-8: Succesfully created PLC IO Table in CX-Programmer

Then, double click the **CJ1W-ETN21 (ETN21 Mode) (Ethernet Unit)**, so that the **CJ1W-ETN21 (ETN21 Mode) [Edit Parameters]** dialog will appear. Make the configuration as demonstrated in Figure 8-8, then left click the **Transfer (PC to Unit)** button, and the PLC will restart.

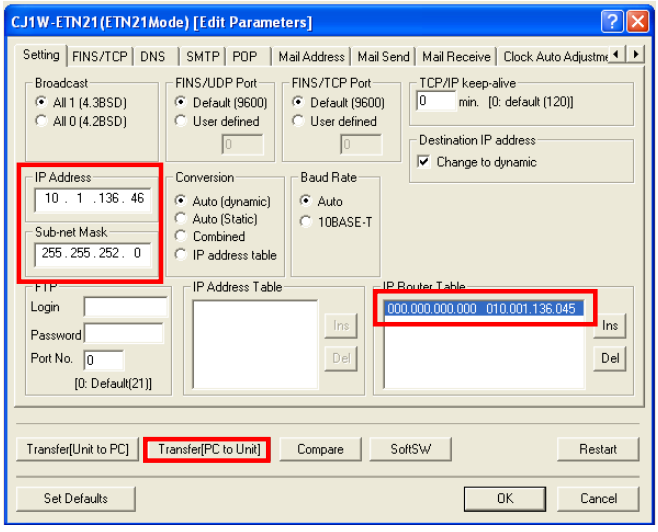


Figure 8-9: CJ1W-ETN21 Configuration in CX-Programmer

If the error alarm indicator in the PLC is light up, restart the PLC by turn off and turn it on again. Once the error alarm is off, left click the button again to disconnect the connection. Then right click the NewPLC1 in the project window and select **Change** in order to change the network type to **Ethernet (FINS/TCP)**, as illustrated in Figure 8-9.

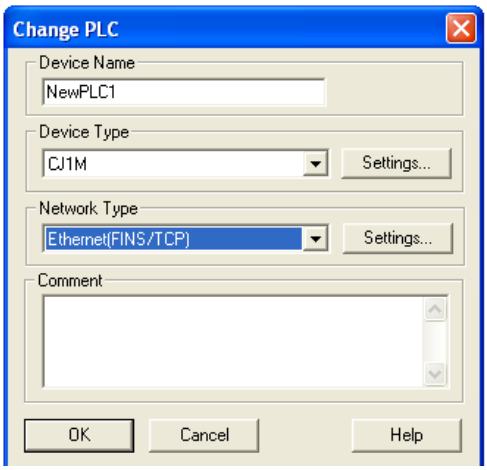


Figure 8-10: Change PLC dialog in CX-Programmer [Ethernet(FINS/TCP)]

Next, left click the **Setting** button, and the **Network Settings [Ethernet (FINS/TCP)]** dialog will shows up. Change the FINS destination Address according to the decimal number set in the PLC node number, and this number determine the IP address of the PLC. Then left click the **Driver** tab and insert the IP address. Once it is completed, left click the **OK** button to close the dialog, Figure 8-110 and Figure 8-11.

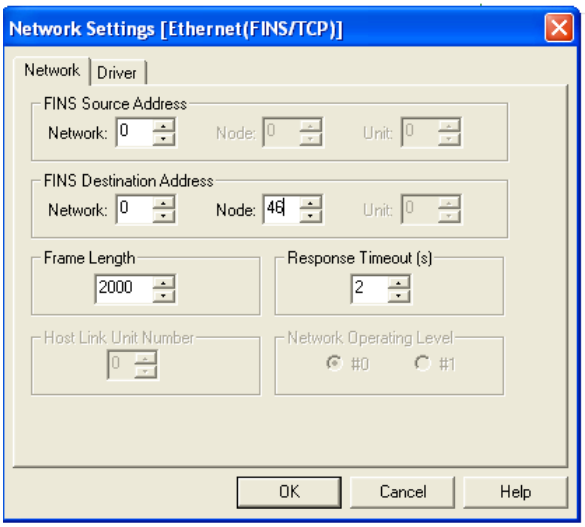


Figure 8-12: Network Settings Part 1 of Ethernet FINS/TCP in CX-programmer

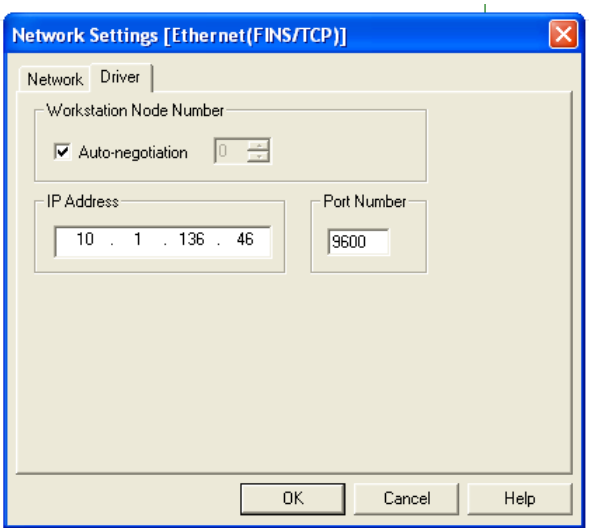


Figure 8-13: Network Settings Part 2 of Ethernet FINS/TCP in CX-programmer

Now the Ethernet configuration in CX-programmer had been done. In order to test whether it is working properly, connect the Ethernet cable from the PLC to the router Ethernet port, and the **LNK** indicator light in the PLC will light up. Click the  button to connect the computer to the PLC. If it is connected, no errors will shows up.

Additionally, user can test the PLC and computer connection by using Command Prompt. Firstly launch the Command Prompt by left click **Start >> All Programs >> Accessories >> Command Prompt**. Then in command prompt, type "ping <PLC IP address>". If the Ethernet configuration had been correctly configured, the output of the pinging should be appearing as the result illustrated in Figure 8-12.

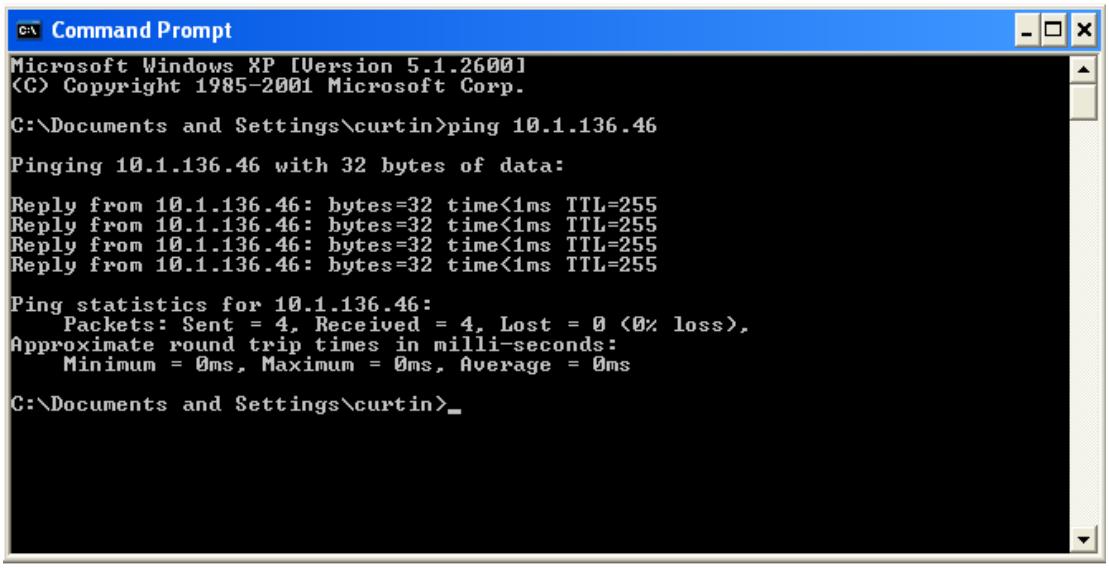


Figure 8-14: Testing Connectivity using Command Prompt in Window XP

Otherwise troubleshoot the problem by refer back to the step by step approach all over again.

Appendix B: Setting Up NI OPC Server

This section demonstrates the step by step configuration and setup of the NI OPC Server. Before performing the configuration below it is important to ensure that the LabVIEW Datalogging and Supervisory Control Module had been installed properly in the computer. The demonstration in this chapter is performing under the Windows Vista Basic operating system, and the LabVIEW used is LabVIEW 2009. Therefore the explanation might be slightly different if the users are using other operating system in their computer.

Firstly, launch the NI OPC Server by left click **Start >> All Programs >> National Instruments >> NI OPC Servers >> NI OPC Servers**. Then left click the **File** tab and select the **New** option. Double click the "*Click to add a channel*" link and the **New Channel – Identification** will show up, as illustrated in Figure 8-15.

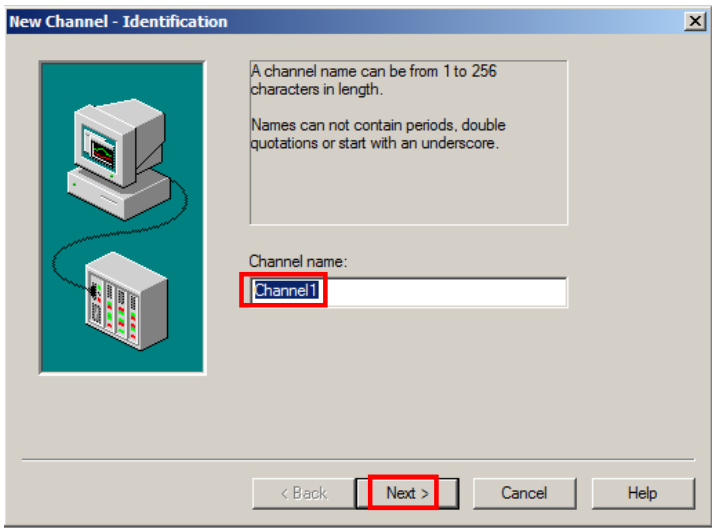


Figure 8-16: New Channel naming in NI OPC Servers

Left click **Next** button to continue. Select the device driver to **Omron FINS Ethernet**, as demonstrated in Figure Figure 8-4.

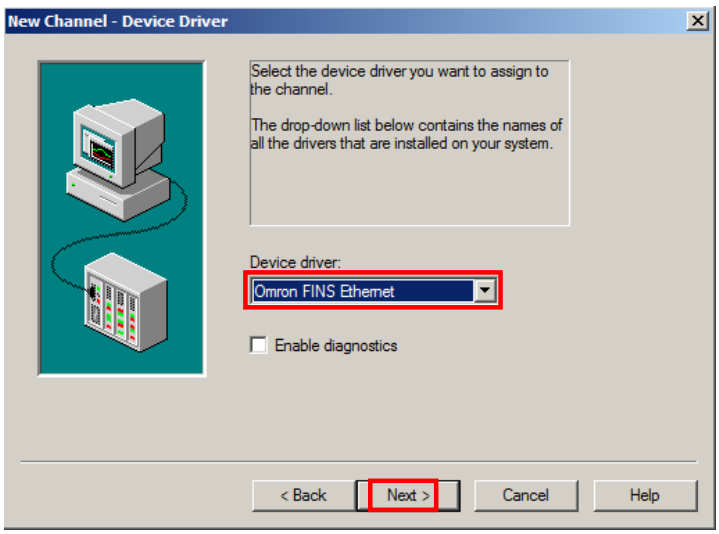


Figure 8-17: Device driver selection in NI OPC Servers

Left click **Next** button to continue. Let the Network Adapter as default, then click **Next** button again. Let the Optimisation Method to be default, which is **Write only latest value for all tags** and the Duty Cycle is 10 writes for every 1 read. Click **Next** button and let the port number remain as 9600. Click **Next** button again and follow by **Finish** button to close this window, as demonstrated in Figure 8-15. Now the channel had been successfully setup.

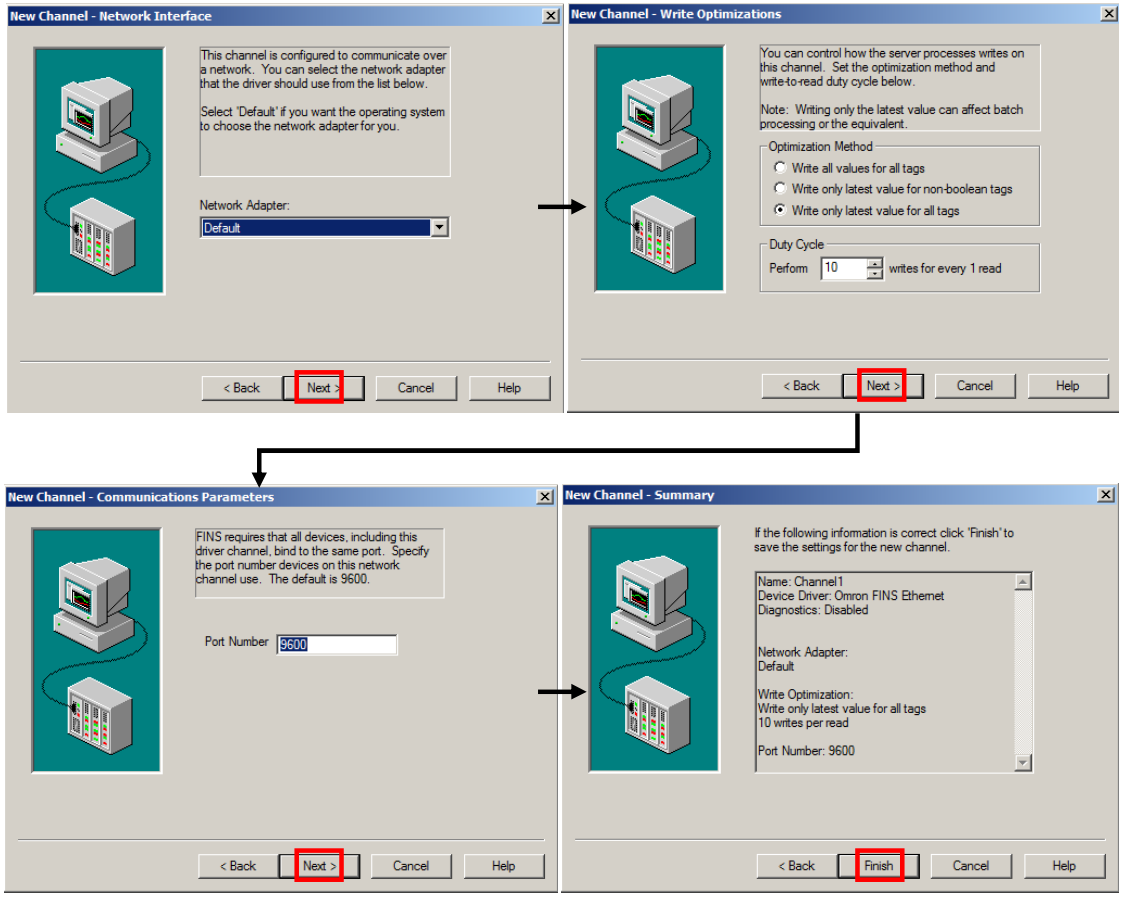


Figure 8-18: Demonstration of New Channel setting up in NI OPC Servers

After that, create the device by left click the **Click to add a device** link. Name the device depend on the user preference, which in this project, the name is **CJ1M-CPU11-ETN21**. Then perform the configuration as illustrated in Figure 8-16. The **Device model** is CJ1 series. Since the IP address of the PLC is 10.1.136.46, the device ID should be identical with it, and the **Destination Node Number** will be 46. Meanwhile as the IP address of the user computer is 10.1.136.47, the **Source Node Number** will be 47.

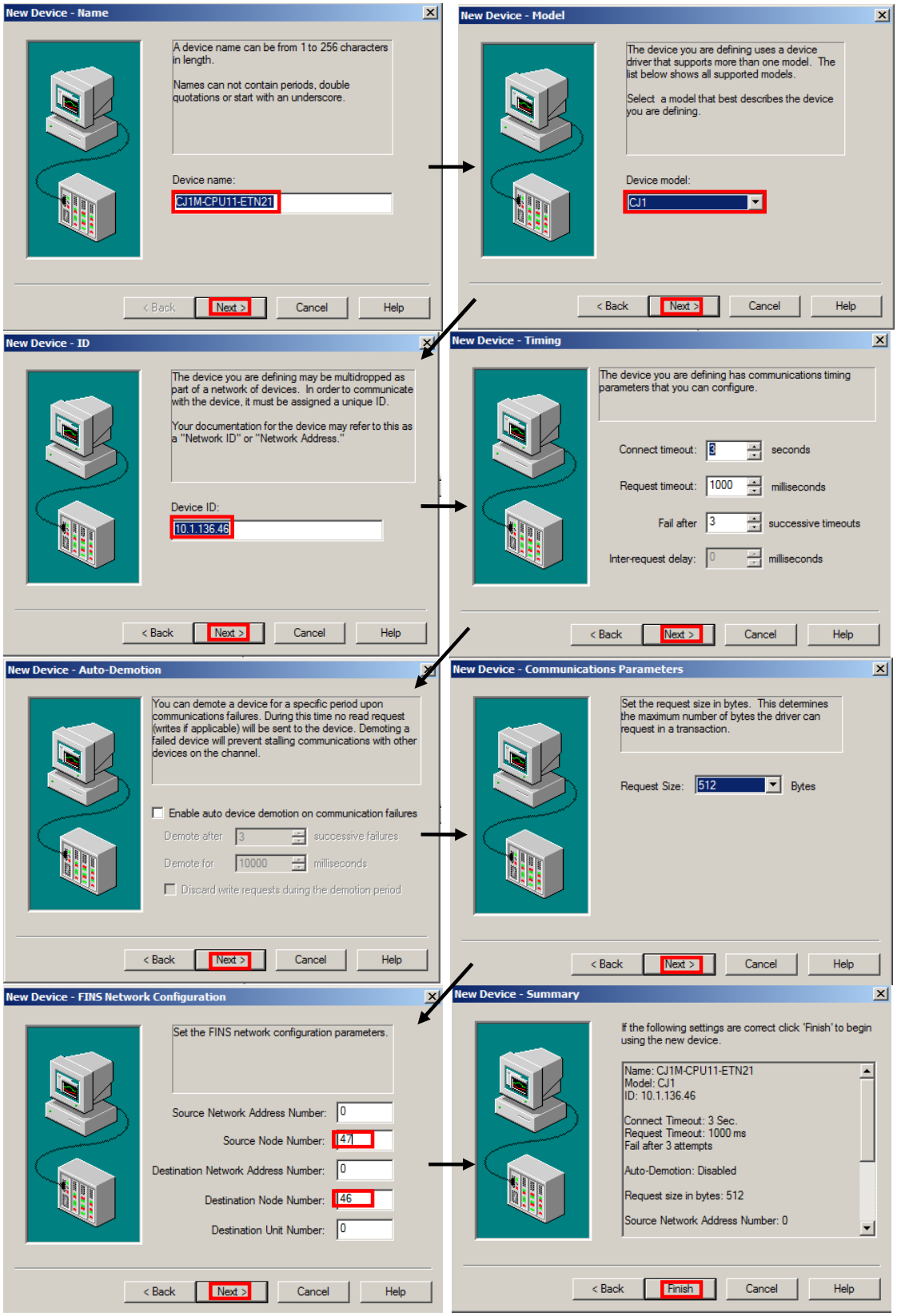


Figure 8-19: Demonstration of creating New Device in NI OPC Servers

Now the device had been successfully configured. Next create the tags by left click the **Click to add a static tag** link, and the **Tag Properties** dialog below will appear. Perform the configuration as demonstrated in Figure 8-17.

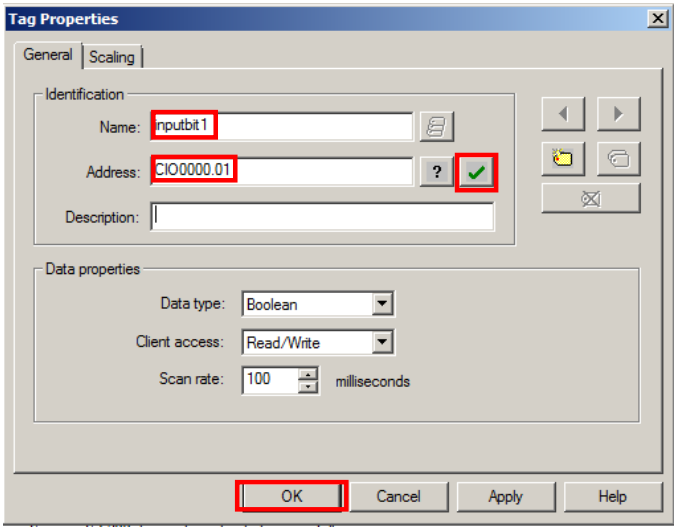


Figure 8-20: Create Tags in NI OPC Servers

Once the configuration is complete, the tag and its details will appear in the table, as illustrated in Figure 8-18.

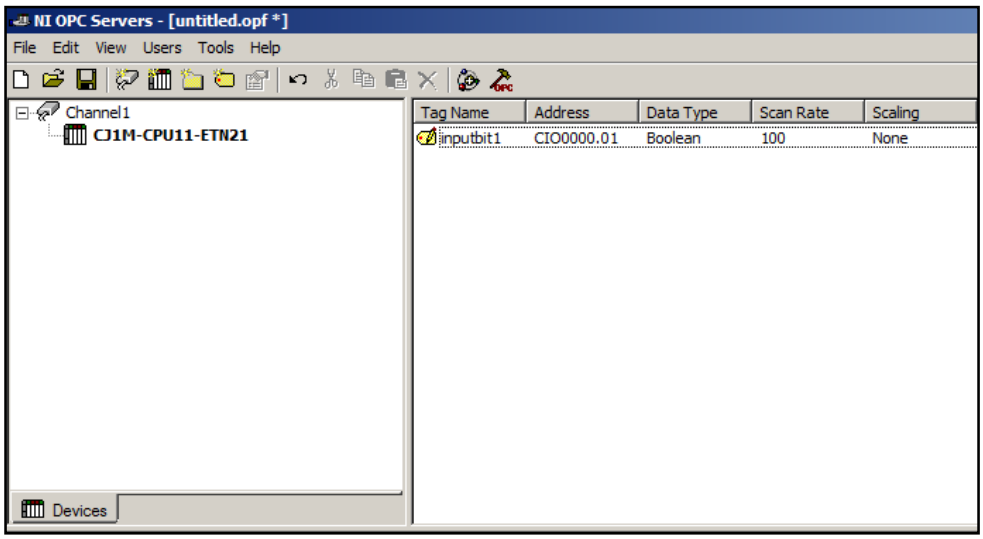


Figure 8-21: NI OPC Servers with a input OPC tag

Launch the **OPC Quick Client** by left click the  icon. It is necessary to launch the **OPC Quick Client** so that the tags can be browsed by the LabVIEW. If the connection between the user computer and PLC is successful, the outcome of the **OPC Quick Client** should be identical with the Figure 8-19, which is the part crop from the demonstration **OPC Quick Client**.

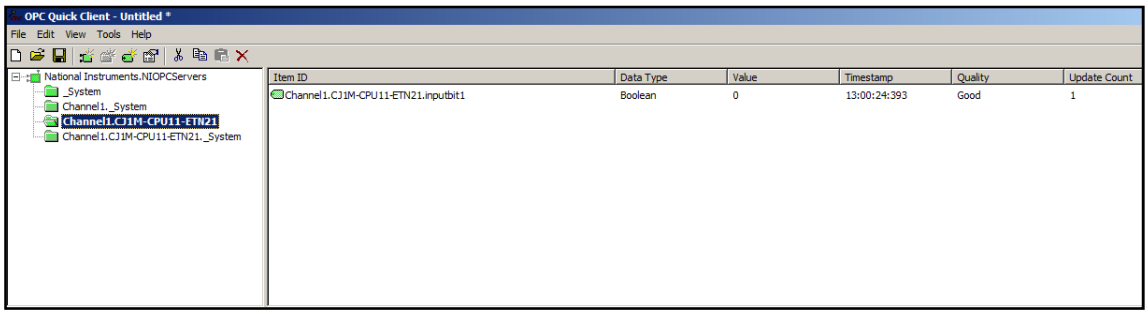


Figure 8-22: NI OPC Quick Client

Now the demonstration of the NI OPC Configuration had been completed. The OPC tags are now can be used in the LabVIEW.

Appendix C: Setting Up LabVIEW Shared Variables

This section demonstrates the step by step configuration and setup of the shared variable in LabVIEW. Before performing the configuration below it is important to ensure that the LabVIEW Datalogging and Supervisory Control Module had been installed properly in the computer. The demonstration in this section is performing under the Windows Vista Basic operating system, and the LabVIEW used is LabVIEW 2009. Therefore the explanation might be slightly different if the users are using other operating system in their computer.

Firstly, launch the LabVIEW software by click **Start >> All Programs >> National Instruments >> LabVIEW 2009 >> LabVIEW**. Once the LabVIEW window had been launched, the **Getting Started** window will appear. Under the **New** title, left click the **Empty Project** to open a new project. The new **Project Explorer** will show up. Next, under the **Item** tab, right click the "*My Computer*" icon and select **New >> I/O Server**, which had shown in the Figure 1 below (left side).

Now left click the **I/O Server** and then the **Create New I/O Server** window will appear. There are many types of I/O server under the **Create New I/O Server** window, but in this project the scope will only focus on the selection of OPC Client as the objective of the configuration is to allow the user computer to connect to the PLC by bounding with OPC tags. So under the **Create New I/O Server** window, left click the "*OPC Client*", and press **Continue** to proceed, Figure 8-20.

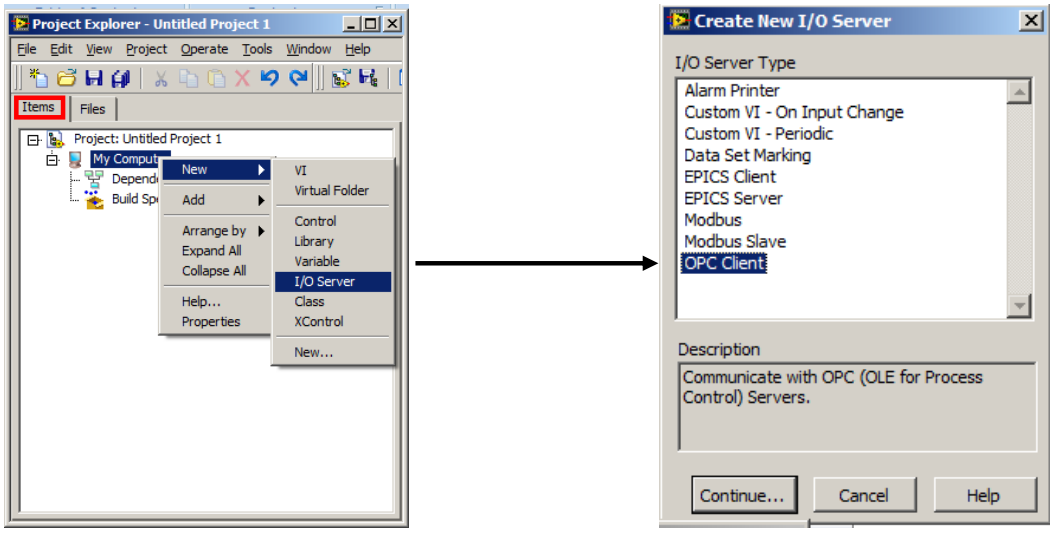


Figure 8-23: Creating New I/O Server (OPC Client) in LabVIEW

Next, the **Configure OPC Client I/O Server** window will show up. So under the **Settings** tab, change the **Update rate (ms)** into 100, and under the **Registered OPC servers** left click the "*National Instruments.NIOPCServers*", follow by left click the **OK** button, Figure 8-21. Once this step had been performed, the "*Untitled Library 1*" will appear in **Project Explorer** window. Expand it and the "*OPC1*" will turn up.

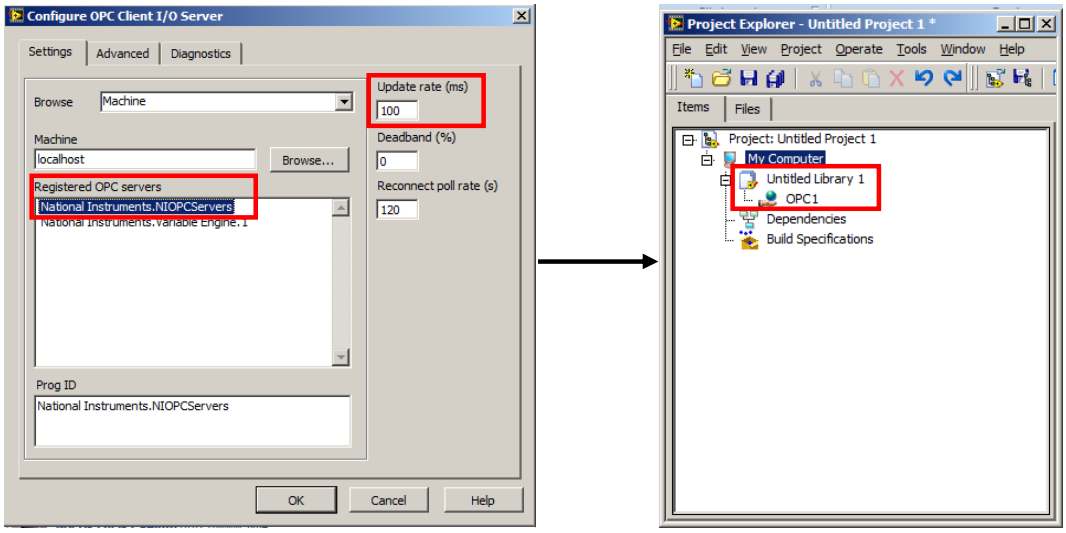


Figure 8-24: Configuration of OPC Client I/O Server

After that, create a new library in the **Project Explorer** window, under the "*My Computer*" icon. This step can be performed by right click the "*Untitled Library 1*" and select **New >> Library**, which is illustrated in Figure 8-22 (left). Next, create the bound variables by right click the newly created library called "*Untitled Library 2*" and select **Create Bound Variables** option, as illustrated in Figure 8-22 (right).

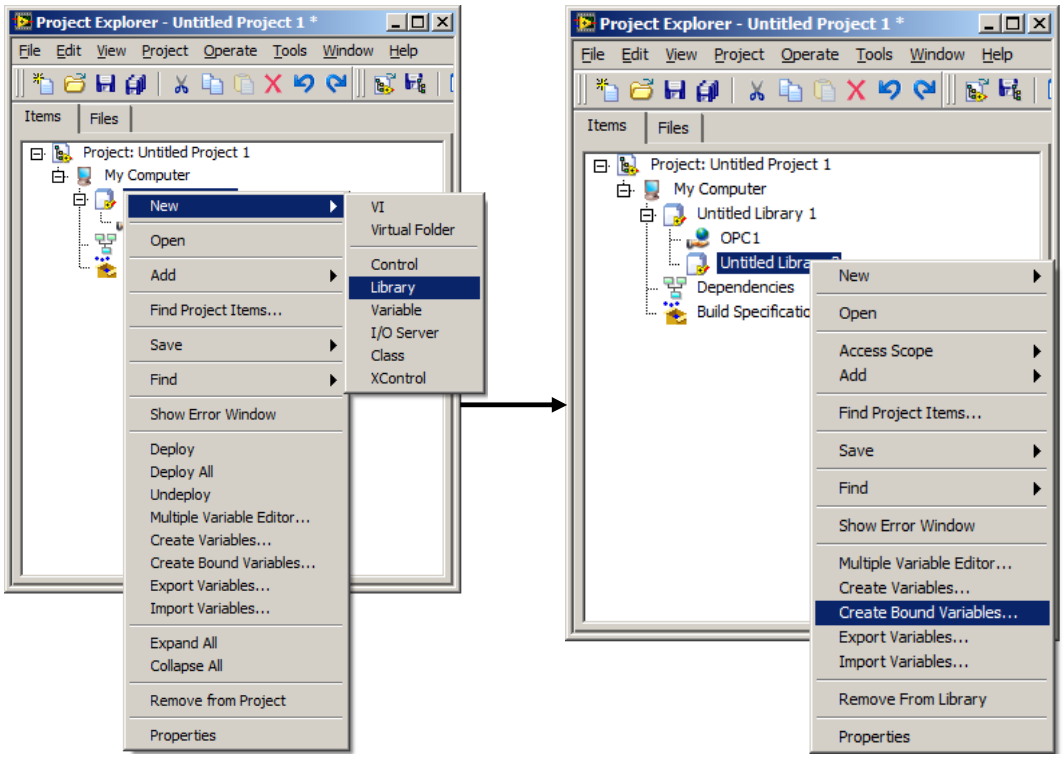


Figure 8-25: Creating New Library and Bound Variables in LabVIEW

Once the previous step had been performed, the **Create Bound Variables** window will show up. At the left hand side of the window, expand all the icons until the relevant OPC tag shown. Then left click the **Add** button in the middle of the window to add all the OPC tag into "Added variables" list, follow by left click the **OK** button. This step had illustrated in Figure 8-23.

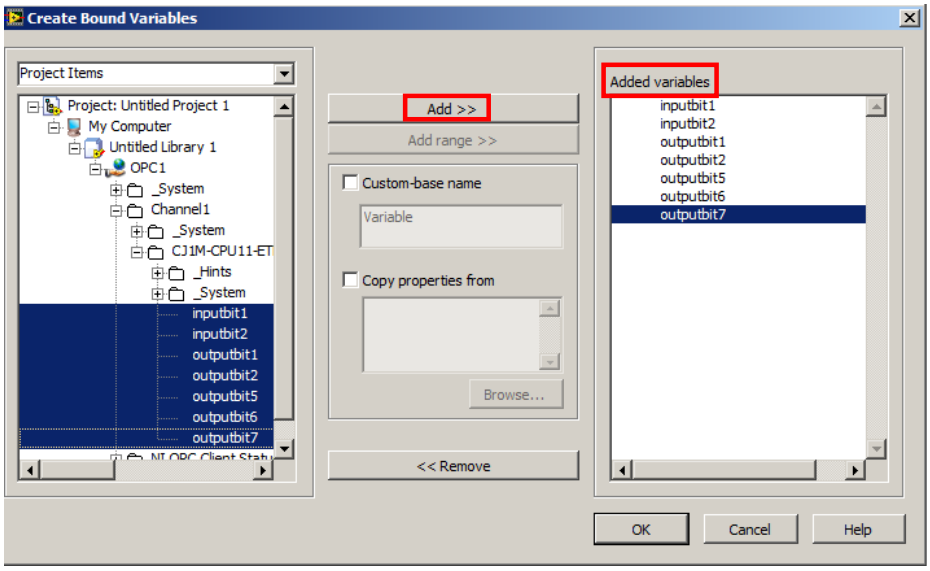


Figure 8-26: Add variables in Create Bound Variables dialog in LabVIEW

Next, the **Multiple Variable Editor** window will pop up. From this window, the details of the variables are clearly shown. Refer to the Figure 8-24 for further details. Then left click the **Done** button to close the window.

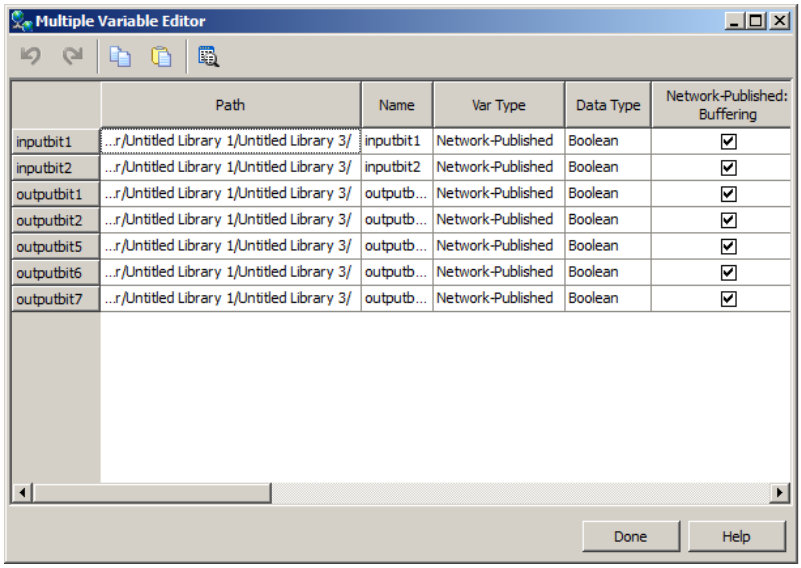


Figure 8-27: Multiple Variable Editor window

Once the shared variables had been successfully created, the list of OPC tags will be shown in the **Project Explorer** window under the "*Untitled Library 3*" library, as illustrated in Figure 8-25.

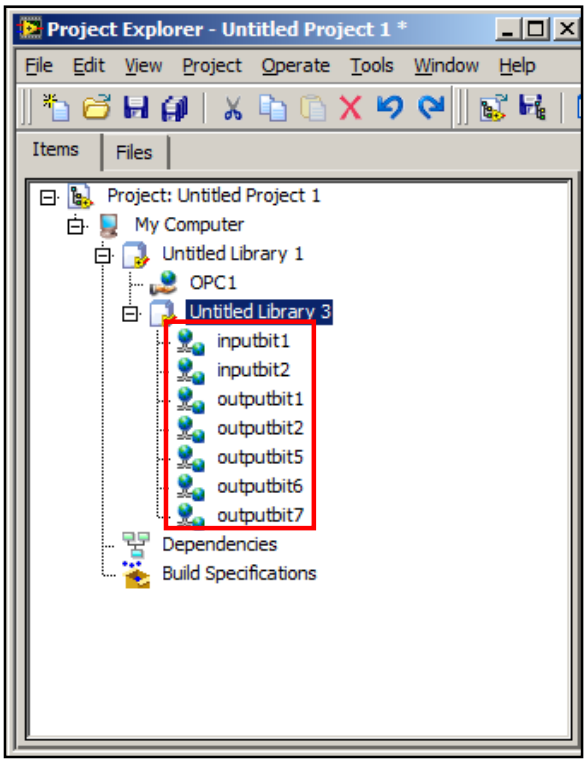


Figure 8-28: Project Explorer with shared variables in LabVIEW

In order to use the shared variable in the VI, firstly create the new VI by right click on the "*My Computer*", and then select **New >> VI**, then **Front Panel** window will pop up.

Then drag the relevant shared variable into the **Front Panel** window. For example in this case the "*inputbit1*" had been dragged and put into the **Front Panel**. Do notice that the "*inputbit1*" appear as **push button** in the **Front Panel**, as illustrated in Figure 8-26.

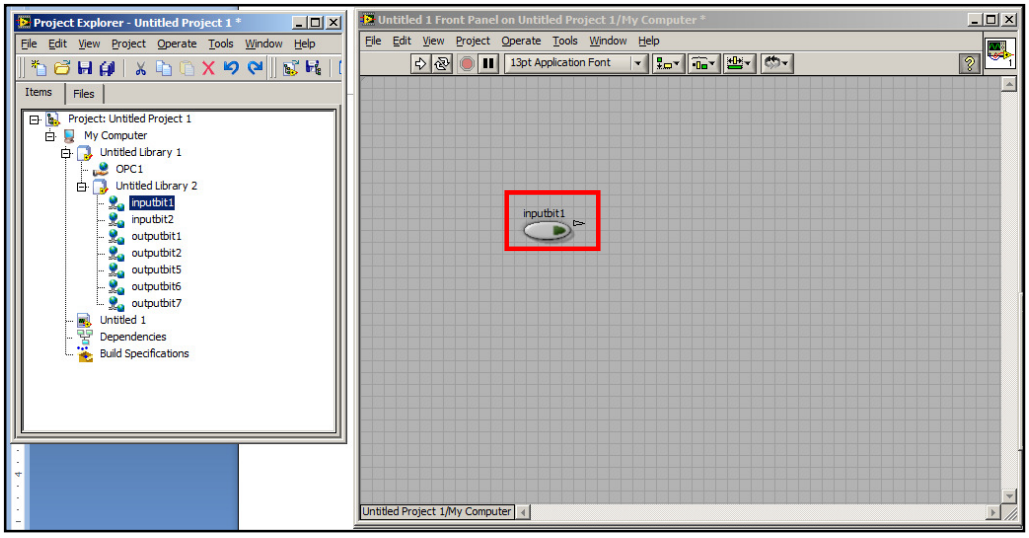


Figure 8-29: VI Front Panel with shared variable push button in LabVIEW

In order to open the **Block Diagram** window, press **Ctrl+E** in keyboard. The "*inputbit1*" appeared as Boolean input in **Block Diagram**, as shown in Figure 8-27.

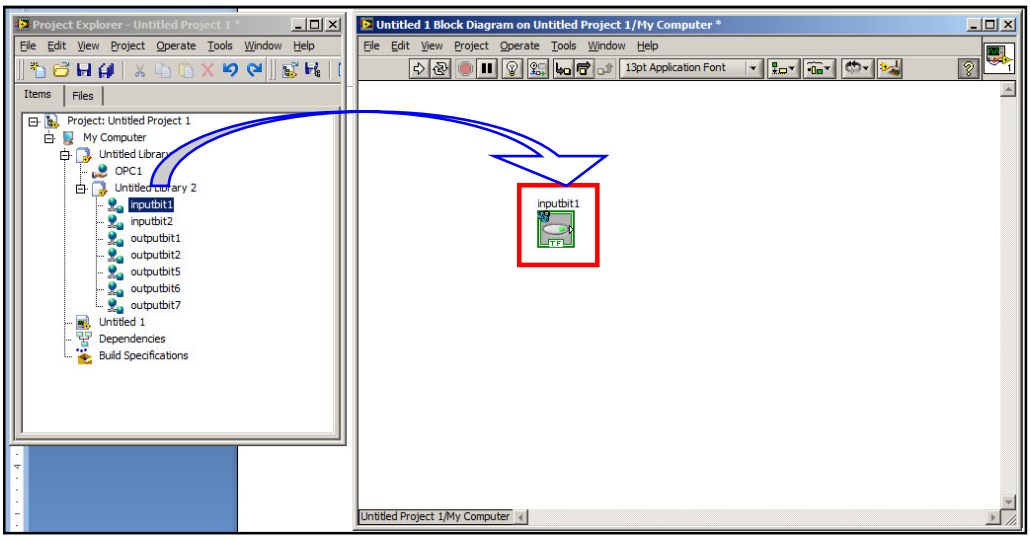


Figure 8-30: VI Block Diagram with shared variable Boolean input in LabVIEW

Next, with the purpose of controlling the output of the PLC, the "*Outputbit1*" need to be used, by drag the "*Outputbit1*" icon from **Project Explorer** window into **Block Diagram** window, which is illustrated in Figure 8-28.

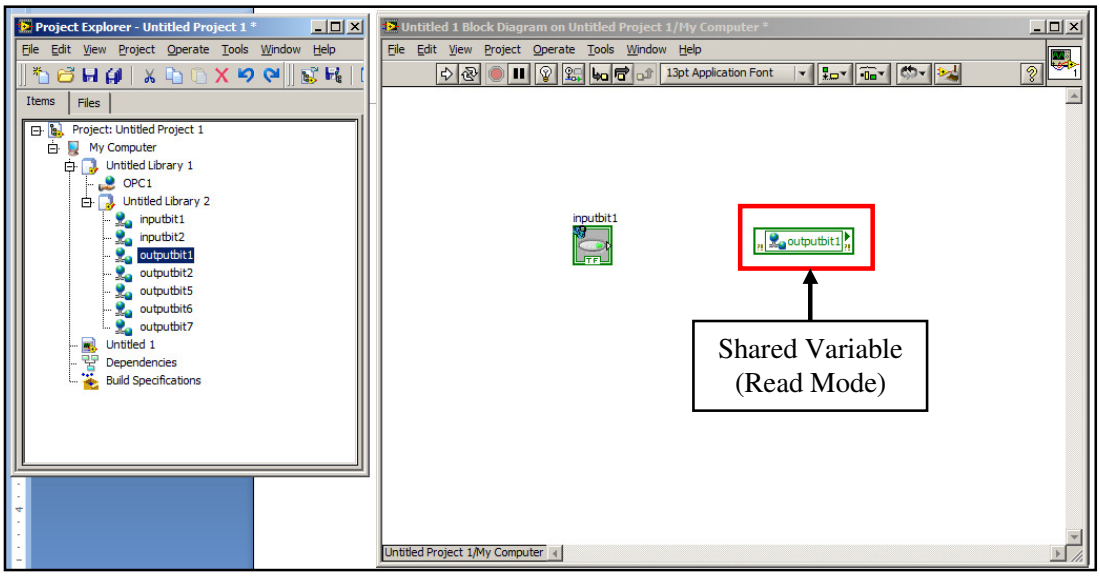


Figure 8-31: VI Block Diagram with input and output shared variables in LabVIEW

For now the "*Outputbit1*" is in Read mode, in order to control the PLC the "*Outputbit1*" must be in Write mode. Therefore in order to change the mode, right click the "*Outputbit1*" and select **Change To Write** option. Notice that once the previous step had been performed, the "*Outputbit1*" icon had changed, as shown in Figure 8-29.

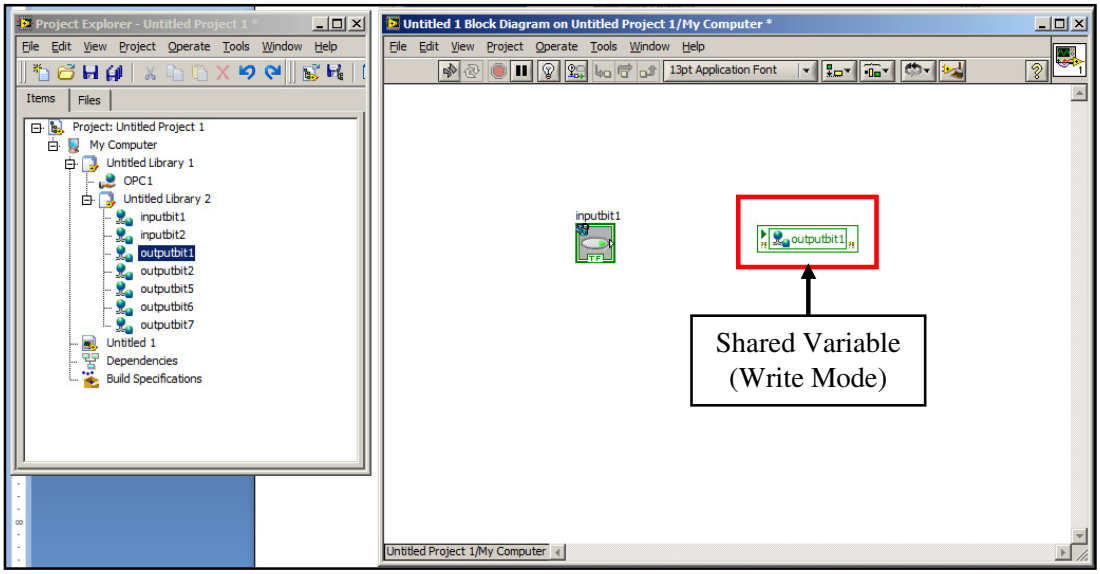


Figure 8-32: Change the output shared variable into write mode in LabVIEW

Now user can draw a wire from "*Inputbit1*" into "*Outputbit1*" without showing any error, as illustrated in Figure 8-30.

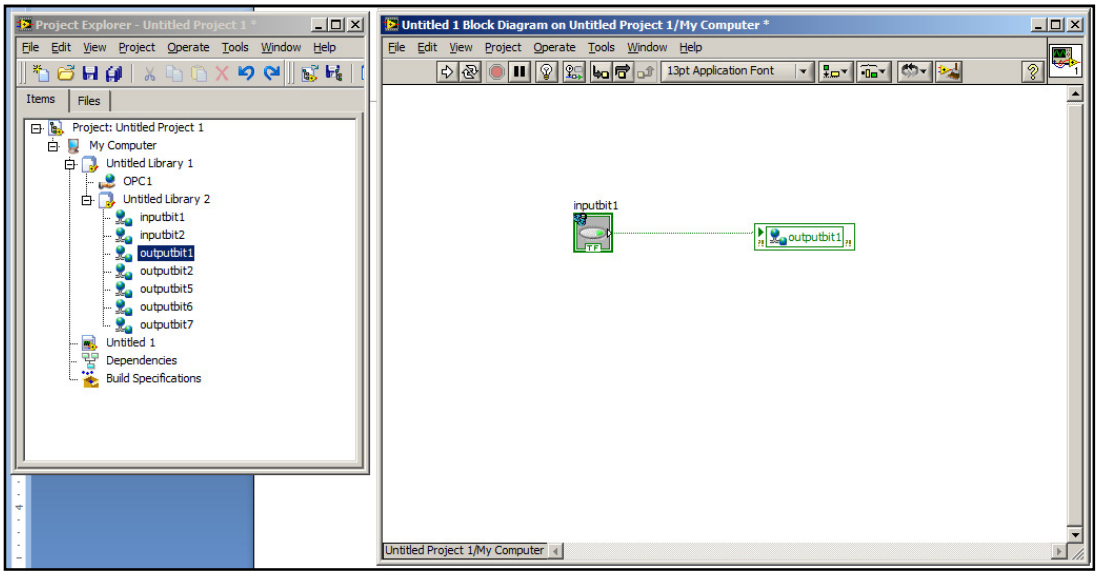


Figure 8-33: Wire the input and output shared variable in LabVIEW

Next, in order to ensure the program run continuously until the user pressed the stop button, a “While Loop” execution control function is needed. In LabVIEW, this can be done by simply right click anywhere in the **Block Diagram** window, and the **Function Palette** will appear. In the **Function Palette**, under the **Express** tab, choose **Execution Control >> While Loop**, as demonstrated in Figure 8-31.

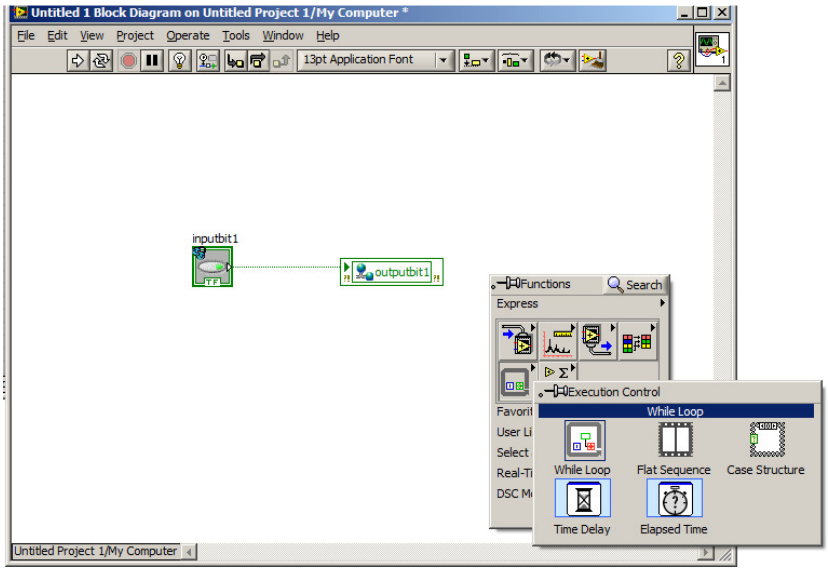


Figure 8-34: Create While Loop in LabVIEW Part 1

After that, drag the mouse from left side to cover the whole program, as demonstrated in Figure 8-32.

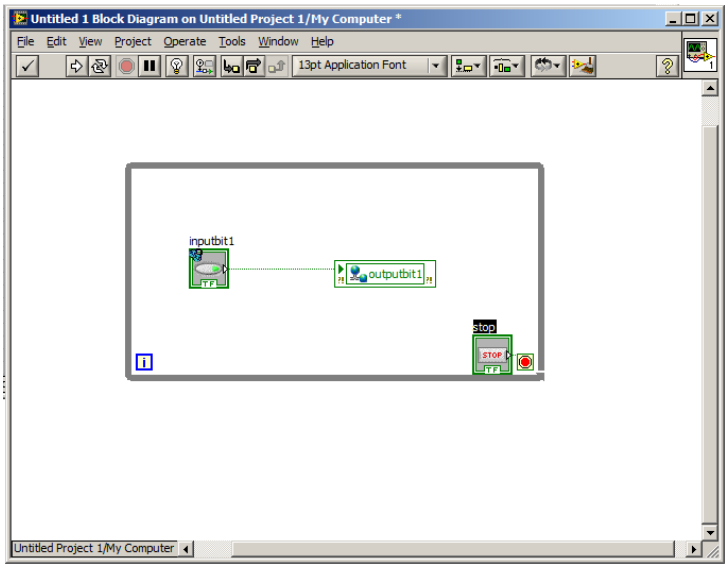


Figure 8-35: Create While Loop in LabVIEW Part 2

Now the demonstration of using shared variable in LabVIEW had done. User can test the program by run the software and control it in **Front Panel**. If the program is correctly configured, output light of the PLC will turn on when the input had been triggered, and the small triangle of the input push button will light on when the VI is running, which indicate the connection between the computer and PLC is valid. Otherwise troubleshoot the configuration by ensure all the steps above had been followed correctly, and the NI OPC Servers configuration, especially IP address, had been configure correctly.

Appendix D: Hardware Specification

OMRON CJ1M-CPU11-ETN21 Programmable Logic Controller (PLC)

Name	Model	Specifications
CJ1M CPU Unit	CJ1M-CPU11-ETN21	I/O bits: 160, Program capacity: 5 Ksteps Data Memory: 32 Kwords (DM: 32 Kwords, EM: None)
CJ-Series Power Supply Unit	CJ1W-PA202	100 to 240V AC, Output Capacity: 2.8A at 5V DC
DC Input Unit	CJ1W-ID211	Terminal block 24V DC, 16 inputs
Relay Output Unit	CJ1W-OC211	Terminal block 250V AC, 0.6A, 8 points
End Cover	CJ1W-TER01	Must be connected to the right end of the CPU Rack. A fatal error will occur if the End Cover is not connected.

Table 8-1: Specifications for each unit in CPU Rack

TECO Motor Type AEEB Squirrel Cage Three Phase Induction Motor

Code	AEEB KB	
Number of Poles	4	
Power	½ HP	
	0.37 kW	
Frequency	50 Hz	
Synchronous Speed	1400 rpm	
Insulation Type	F	
Ambient Temperature	40°C	
Input Voltage	220 – 240 V	Delta
Input Current	2.02 – 1.85 A	
Output Voltage	380 – 415 V	Wye
Output Current	1.17 – 1.07 A	

Table 8-2: The Specification of the Three Phase Squirrel Cage Induction Motor TECO

Appendix F: Circuit Diagram of PLC and Variable Frequency Drive

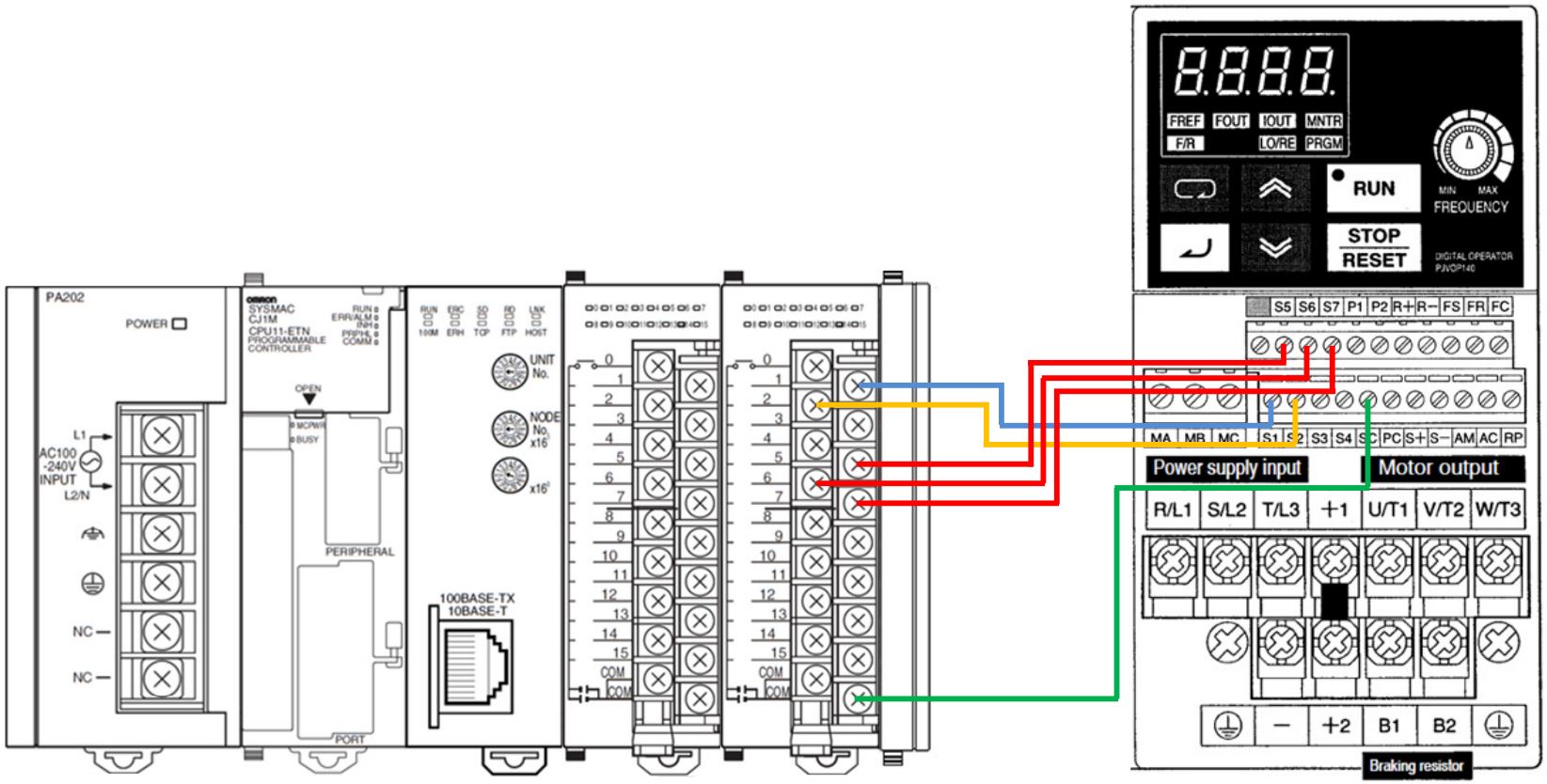


Figure 8-36: Circuit Diagram of PLC and Variable Frequency Drive

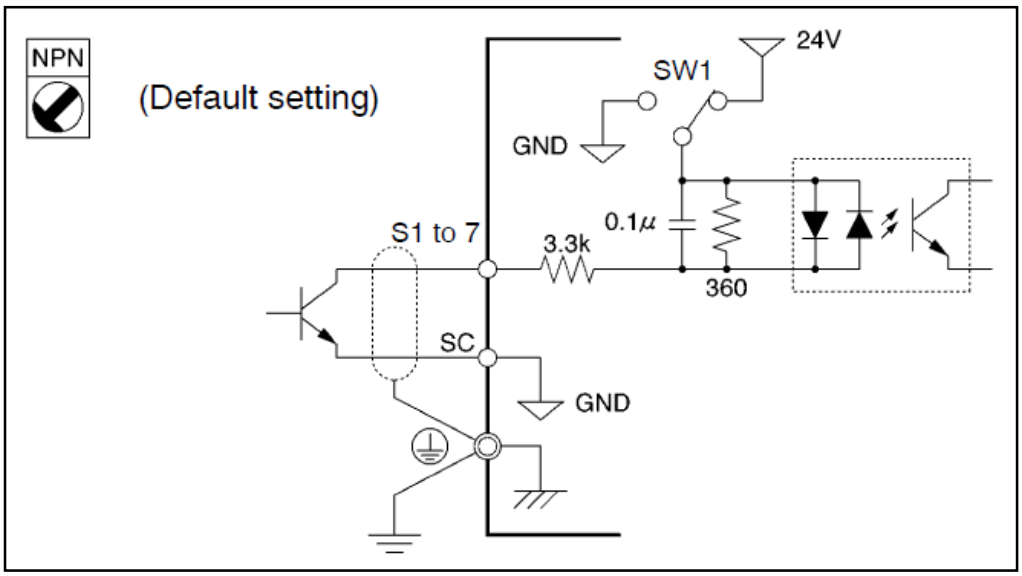


Figure 8-37: Internal Circuit of the 3G3MV Inverter